Tesis Doctoral

Arquitectura de Agente Emocional para Aplicaciones de Control en Tiempo Real

Autor

Carlos Domínguez

Directores

Dr. Houcine Hassan

Dr. Alfons Crespo

Valencia, 5 de julio de 2017

a mis padres

Agradecimiento

Agradezco la ayuda de mis compañeros de departamento, por sus ideas constructivas y por su apoyo; en especial a mis tutores, los profesores *Alfons Crespo* y *Houcine Hassan*.

Agradezco también la paciencia que mi familia ha tenido conmigo, cuando por razones de trabajo, y en tantas ocasiones, he restado tiempo de estar con ellos.

Resumen

Los agentes artificiales constituyen una tecnología de apoyo para la resolución de problemas. Un agente es un sistema con un grado significativo de autonomía, lo que le permite descargar a su usuario de tareas que éste no puede o no quiere realizar. Aun siendo autónomo en sus comportamientos, el agente asume los objetivos de su usuario como propios, ya que existe un contrato entre el agente y su representado. Se trata de una tecnología potente y que interesa desarrollar, con lo que el área de investigación en agentes está abierta y hay un esfuerzo continuo para construir agentes con cada vez mejores prestaciones.

Siendo los agentes sistemas complejos, resulta necesario definir marcos de desarrollo que permitan concebirlos, diseñarlos y construirlos. Conocemos a estos marcos como arquitecturas de agentes artificiales. Cada una de estas arquitecturas se caracteriza por ciertas ideas clave, relacionadas con la forma en que el agente representa su conocimiento y organiza su comportamiento, en lo que se denomina un paradigma. Sin duda, queda mucho recorrido en este campo - ampliando por ejemplo las áreas de aplicación, o permitiendo funcionalidades adicionales, o aumentando la eficiencia de los procesos implicados, tanto en lo relativo al comportamiento del agente cuando éste está en explotación, como durante el propio proceso de construcción y validación del sistema.

En este trabajo se propone una arquitectura de agente artificial en el que la organización del comportamiento está dirigida por un proceso emocional. Se trata de una arquitectura bio-inspirada. La emoción en este caso, sin embargo, es una versión muy simplificada del proceso emocional en los agentes emocionales naturales.

Aunque se han definido otras arquitecturas de agentes artificiales basadas en emociones, han sido enfocadas, sobre todo, a construir agentes con habilidades sociales; normalmente para interactuar con las personas. Posiblemente esto ha sido debido a que ya hace mucho que se aceptaba la importancia de la emoción en las relaciones sociales entre los seres humanos; cuando éstos interpretan el estado interno de los demás o expresan su propio estado, alterando con ello sus comportamientos.

Sin embargo, el papel fundamental de la emoción en un amplio espectro de procesos cognitivos está siendo reconocido a raíz de la investigación en psicología y neurología. Las emociones parecen contribuir de forma esencial en procesos como la percepción, el

aprendizaje, la memoria, la toma de decisiones o la resolución de problemas. El propio pensamiento deliberativo racional estaría dirigido por las emociones.

Teniendo en cuenta esta nueva visión de las emociones, en este trabajo se ha investigado el rol de la emoción en los procesos cognitivos de un agente artificial relacionados con la toma de decisiones en general, no sólo en lo relativo a los procesos de interrelación social. Así por ejemplo, en la aplicación considerada como caso de estudio de este trabajo, el agente emocional controla una plataforma de robot móvil de servicio, en la que no hay una capa de comportamiento social importante, y donde los procesos emocionales motivan fundamentalmente los comportamientos relacionados con problemas surgidos en un entorno físico, con objetos, piezas, o espacios de operación y navegación.

En esta tesis se define una especificación para la arquitectura de agente emocional artificial propuesta y se discute aspectos de implementación de dicha arquitectura.

Abstract

Artificial agents are a technology suitable for solving problems. Agents can perform tasks that their users cannot and/or do not want to accomplish. Agents are systems with a significant degree of autonomy. Even being autonomous in their behavior, they assume the users' goals as their own goals, because there is an agreement between the agent and the user. It is a powerful technology, and the research on this field is very active.

As agents are complex systems, it is necessary to define development frameworks that facilitate their conception, design and construction. We name these frameworks, artificial agent architectures. Each architecture is characterized by a few key ideas related to the way the agent represents its knowledge about the world, and how it organizes its behavior. We call these key ideas a paradigm.

In this work, an artificial agent's architecture is proposed. In this architecture the organization of the behavior is emotionally driven. It is a bio-inspired architecture. The emotion in this case, however, is a very simplified version of the emotional process in the natural emotional agents.

Although other agent architectures based on emotions have been proposed, they have been usually focused on the social skills of the agents, normally to interact with people. This situation could have been caused due to the knowledge we had about the importance of the emotion in the social relations between human beings, when people recognize the internal state of the others, or show their own internal states, and the emotional communication influences their behavior.

However, the fundamental role of the emotion in a wide range of cognitive processes is being recognized because of the recent research in psychology and neuroscience. Emotions seem to make an essential contribution in processes such as perception, learning, memory, decision-making and problem solving. Deliberative rational thoughts themselves would be directed by emotions.

Given this new view about the emotion, in this thesis, we have investigated the role of the emotions in the cognitive processes of an artificial agent, related them to the general decision making problem, not just the social interaction problem. As an example, in the application considered as a case study in this project, the emotional agent controls a mobile robot platform, in which there is not an

important behavior layer of social interaction, and the emotional processes primarily motivate behaviors related to problems in a physical environment, with objects, parts, or areas of operation and navigation.

In this thesis, we have defined a specification for the proposed emotional agent architecture, and have discussed the implementation aspects of it.

Resum

Els agents artificials constitueixen una tecnologia de suport per a la resolució de problemes. Un agent és un sistema amb un grau significatiu d'autonomia, el que li permet descarregar al seu usuari de tasques que aquest no pot o no vol fer. Fins i tot sent autònom en els seus comportaments, l'agent assumeix els objectius del seu usuari com a propis, ja que hi ha un contracte entre l'agent i el seu representat. Es tracta d'una tecnologia potent i que interessa desenvolupar, de manera que l'àrea de recerca en agents està oberta i hi ha un esforç continu per construir agents amb cada vegada millors prestacions.

Sent els agents sistemes complexos, resulta necessari definir marcs de desenvolupament que puguen permetre concebre'ls, dissenyar-los i construir-los. Coneixem a aquests marcs com arquitectures d'agents artificials. Cadascuna d'aquestes arquitectures es caracteritza per certes idees clau, relacionades amb la forma en què l'agent representa el seu coneixement i organitza el seu comportament, en el que s'anomena un paradigma. Sens dubte, queda molt de recorregut en aquest camp - ampliant les àrees d'aplicació, o permetent funcionalitats addicionals, o augmentant l'eficiència dels processos implicats, tant pel que fa al comportament de l'agent quan aquest està en explotació, com durant el mateix procés de construcció i validació del sistema.

En aquest treball es proposa una arquitectura d'agent artificial en què l'organització del comportament està dirigida per un procés emocional. Es tracta d'una arquitectura bio-inspirada. L'emoció en aquest cas, però, és una versió molt simplificada del procés emocional en els agents emocionals naturals.

Tot i que s'han definit altres arquitectures d'agents artificials basades en emocions, han estat enfocades, sobretot, a construir agents amb habilitats socials; normalment per interactuar amb les persones. Possiblement això ha segut perquè ja fa molt que s'acceptava la importància de l'emoció en les relacions socials entre els éssers humans; quan aquests interpreten l'estat intern dels altres o expressen el seu propi estat, alterant amb això els seus comportaments.

No obstant això, el paper fonamental de l'emoció en un ampli espectre de processos cognitius està sent reconegut arran de la investigació en psicologia i neurologia. Les emocions semblen contribuir de forma essencial en processos com la percepció,

l'aprenentatge, la memòria, la presa de decisions o la resolució de problemes. El mateix pensament deliberatiu racional estaria dirigit per les emocions.

Tenint en compte aquesta nova visió de les emocions, en aquest treball s'ha investigat el paper de l'emoció en els processos cognitius d'un agent artificial relacionats amb la presa de decisions en general, no només pel que fa als processos d'interrelació social. Així per exemple, en l'aplicació considerada com a cas d'estudi d'aquest treball, l'agent emocional controla una plataforma de robot mòbil de servei, en què no hi ha una capa de comportament social important, i on els processos emocionals motiven fonamentalment els comportaments relacionats amb problemes sorgits en un entorn físic, amb objectes, peces, o espais d'operació i navegació.

En aquesta tesi es defineix una especificació per a l'arquitectura d'agent emocional artificial proposta i es discuteixen aspectes d'implementació de la arquitectura.

Contenido

Lista de Figuras

Lista de Tablas

1. Introducción

Inspirándose en los agentes emocionales naturales, este proyecto define aRTEA[1], una arquitectura de agente artificial en la que las emociones juegan un papel esencial en el comportamiento del agente.

El acrónimo RTEA – Real Time Emotional Agent, usado para identificar a los agentes construidos siguiendo esta arquitectura, trata de remarcar el rol de la emoción en el comportamiento, así como que los RTEAs son concebidos para afrontar problemas en entornos dinámicos con requerimientos de respuesta en tiempo real.

[1]Notación

En esta memoria:

"aRTEA" se referirá a la *arquitectura* de agente emocional.

"bRTEA" se referirá al *constructor* (builder) del agente.

"uRTEA" al *usuario* del agente.

"RTEA" a un agente definido según aRTEA.

1.1. Objetivo

El objetivo de esta tesis ha sido definir aRTEA, una arquitectura de agente artificial basada en emociones.

Para ello, se ha planteado los siguientes objetivos parciales:

1. Definir una especificación para aRTEA basada en un modelo estructural-funcional que describa los componentes de RTEA y sus interrelaciones.
2. Definir un mecanismo de motivación y atención emocional como núcleo de los procesos operativos en aRTEA.
3. Evaluar el mecanismo de motivación y atención emocional en base a las siguientes características:
    - Es funcional y efectivo; de forma que actúa dirigiendo la atención de RTEA.
    - Es ajustable; con lo que el comportamiento de RTEA está condicionado al estado emocional, que modifica su actitud.
    - Es controlable; con lo que permite definir diferentes caracteres de agente adaptados a diferentes tipos de problemas.
4. Caracterizar computacionalmente el mecanismo emocional para abordar su implementación, y comprobar que el agente es realizable con la tecnología de procesadores actualmente disponible. Proponer alternativas de implementación y evaluarlas respecto a los requerimientos funcionales del agente.

1.2. Contribución de este trabajo

La contribución de este proyecto ha sido la definición y evaluación de los siguientes conceptos:

1. aRTEA – una especificación de arquitectura de agente emocional.
2. iRTEAcore – una implementación de aRTEA, basada en un procesador principal y un coprocesador acelerador del proceso de motivación y atención emocional; habiéndose considerado las siguientes alternativas para el coprocesador:
   - síntesis de un coprocesador específico sobre FPGA – *Field Programmable Gate Array*
   - uso de dispositivos GPU – *Graphics Processor Unit*
   - aprovechamiento de instrucciones SIMD – *Single Instruction, Multiple Data*
   - uso de núcleos en un procesador multicore
3. appRTEAassistant – una aplicación de iRTEAcore, que implementa un agente asistente multifuncional basado en una plataforma de robot móvil. Esta aplicación se ha definido para evaluar la arquitectura.

1.3. Contenido de esta memoria

Esta tesis se presenta como una colección de artículos; de este modo:

- En el capítulo 2 se presenta la arquitectura de agente emocional.
- Los capítulos 3 a 7 recopilan publicaciones relacionadas con distintos aspectos de la especificación de la arquitectura y su implementación.
    - El capítulo 3 presenta la implementación del coprocesador emocional sobre un dispositivo FPGA.
    - El capítulo 4 presenta la implementación del coprocesador sobre un dispositivo GPU.
    - El capítulo 5 trata el uso de instrucciones SIMD para acelerar el proceso de motivación emocional.
    - El capítulo 6 presenta la arquitectura de agente emocional y su implementación en un procesador multicore.
    - El capítulo 7 evalúa la arquitectura de agente emocional para una aplicación de robot móvil de servicio.
- En el capítulo 8 se resume los resultados y se presenta las conclusiones.

2. Arquitectura

En este apartado se describe aRTEA, la arquitectura de agente emocional en tiempo real objeto de este proyecto.

aRTEA incorpora conceptos heredados de otras arquitecturas de agentes artificiales, pero a su vez, propone algunos cambios de enfoque en el diseño de agentes que son distintivos de esta especificación.

Después de un planteamiento de objetivos, se pasa a definir la arquitectura, presentando las características de los RTEAs desde el punto de vista de su comportamiento y de la funcionalidad que ofrece a uRTEA. Se presenta también qué ventajas supone para bRTEA seguir esta especificación.

Continúa con una descripción de la estructura general del agente y sus componentes principales, justificando la descomposición modular propuesta. Sigue después detallando cada componente, describiendo los conceptos y los procesos operativos de cada uno de los módulos que son esenciales en la especificación.

Como caso de aplicación del agente, se considera un robot móvil asistente multifuncional, que ha servido para evaluar el mecanismo de motivación y atención.

2.1. Objetivos de la especificación aRTEA

Una arquitectura de agente artificial debe proveer un marco de desarrollo y gestión; ayudando a constructores y usuarios en las actividades que deben llevar a cabo en cada una de las etapas del ciclo de vida de los agentes. Estas etapas deberían contemplar la concepción del proyecto de aplicación, el diseño, construcción, integración, validación e implantación del agente, su explotación, y también su reciclado, tanto material como de conocimiento; todo esto desde un punto de vista integral y ecológico.

Las características que pudieran identificarse como potenciadoras a la hora de satisfacer los requerimientos de cada una de las fases del ciclo de vida del agente, son criterios objetivos para valorar una arquitectura y aconsejar o desaconsejar su utilización frente a otras alternativas. Sin embargo, una arquitectura da respuesta a un problema complejo, con lo que la valoración de la misma en su conjunto, tiene finalmente que considerar un balance entre las distintas características deseables.

aRTEA es una especificación genérica independiente de la implementación. Se centra sólo en los mecanismos operativos del agente, realizando una descripción conceptual de dichos mecanismos, aunque considera cuestiones importantes que deben tenerse en cuenta en la implementación, en la forma de un modelo computacional.

A partir de los objetivos que se plantearán a continuación, la especificación de aRTEA ha consistido fundamentalmente en definir:

1. Una representación para el conocimiento del agente.
2. Un mecanismo para la organización de su comportamiento, basado en un proceso de motivación y atención emocional.

El diseño ha sido dirigido por una serie de objetivos parciales relativos al marco de desarrollo, al diseño del agente y a su implementación.

2.1.1. Objetivos relativos al marco de desarrollo

La especificación aRTEA ha de servir de marco de desarrollo de RTEAs. Así, desde un punto de vista práctico o de ingeniería, aRTEA debe facilitar el trabajo de bRTEA en la puesta a punto de RTEAs que sean útiles a uRTEA.

Marco de desarrollo

Como marco, ofrecerá una guía que establezca pautas a seguir de forma sistemática en el diseño y construcción de los distintos componentes, estableciendo su estructura y comportamiento, así como las interfaces que deben implementarse para poder integrarse en el sistema.

Marco de integración

Del mismo modo debe permitir adaptar e integrar como componentes de RTEA otras partes recicladas desde aplicaciones de agentes que sigan otros enfoques distintos, y que bRTEA considere conveniente reutilizar. Para esto último, los procesos operativos de aRTEA deben ser suficientemente flexibles para permitir dicha adaptación e integración, aun a expensas de aceptar una merma en sus prestaciones funcionales cuando las partes reutilizadas no sigan el paradigma aRTEA. Dicha adaptación e integración se aplicará fundamentalmente a los componentes de comportamiento del agente.

Debe ofrecer por tanto un modelo que permita la definición de comportamientos adaptados ya desde el origen a la arquitectura aRTEA, pero también debe permitir integrar comportamientos que ya están desarrollados y contrastados en otras aplicaciones previas, e indicar cómo adaptarlos para poder integrarlos.

Granularidad en la integración de comportamientos

La adaptación de los comportamientos para su integración con el mecanismo de motivación y atención de RTEA, debe poderse hacer hasta el nivel de detalle que bRTEA considere oportuno una vez considerado el esfuerzo de adaptación necesario.

Ejemplo sobre la granularidad

Como ejemplo de adaptación e integración de comportamientos, consideremos el problema de la planificación del camino a seguir en el desplazamiento del robot móvil asistente multifuncional.

En este caso se ha reutilizado un comportamiento previamente disponible, basado en un conocido algoritmo de búsqueda de camino mínimo en grafos, el cual se ha aplicado a un grafo de espacios-accesos que modela el espacio de navegación del robot.

La adaptación ha consistido en descomponer el espacio de búsqueda en sub-espacios, mediante un esquema jerárquico, de forma que el problema de la planificación del camino ha podido ser descompuesto en un conjunto de sub-problemas de planificación.

La posibilidad de granularidad variable en la integración de comportamientos en aRTEA, ha permitido que bRTEA pudiera decidir sobre el grado de descomposición del espacio de navegación. Así, la descomposición ha podido realizarse tanto desde el nivel que define la arquitectura del edificio, con sus distintas plantas, alas, o habitaciones, y que define espacios-accesos muy estables en el tiempo, como en el nivel del mobiliario, medianamente estable en el tiempo, o incluso llegando hasta el nivel de la población que opera y se desplaza en el entorno, y que define espacios-accesos muy dinámicos en el tiempo.

Dichas descomposiciones han permitido que cada uno de los sub-problemas de planificación se haya podido integrar en el mecanismo de motivación y atención emocional del RTEA. Con dicha integración, el sistema de atención ha pasado a dirigir el avance del proceso de planificación de camino global, motivando de forma diferente cada una de las alternativas de camino en función de apreciaciones emocionales sobre la situación de los distintos sub-problemas.

Este enfoque de planificación de camino tiene en consideración el conocimiento que el agente va adquiriendo de forma dinámica acerca de su espacio de navegación, ya sea mediante la percepción directa de su espacio local a través de sus sentidos, o mediante otros canales de consulta de información, como por ejemplo sus conversaciones con otros agentes colaboradores que disponen de una visión de otras partes del entorno, o la lectura en una red de sensores remotos.

Así, con la dirección del *Sistema de Atención*, la búsqueda de alternativas de camino se va avanzando sobre los sub-espacios sobre los que se tiene conocimiento con un cierto grado de confianza, y por el que obtienen motivación.

Con esta descomposición del problema y su integración en el mecanismo de motivación y atención emocional, la planificación se puede intercalar con la acción, de forma natural. En contraste, otro enfoque diferente consistiría en realizar primero una búsqueda de un camino óptimo en el grafo de espacios-accesos completo, cuando posiblemente no puedan tenerse en consideración las circunstancias cambiantes en cada lugar del espacio de navegación, y después, comenzar a aplicar el plan. Dependiendo de las circunstancias, podría tener que descartarse gran parte del mismo.

Observación sobre el ejemplo anterior

Este ejemplo, utilizado para la discusión del grado de integración de los comportamientos en el mecanismo de motivación y atención emocional de aRTEA, podría cuestionarse si el coste computacional del cálculo del camino óptimo en el grafo de espacios-accesos completo fuese despreciable, y pudiese asumirse descartar parte de sus resultados en cada ciclo de planificación-acción. Más, cuando el mecanismo de motivación y atención emocional supone un coste computacional adicional que debe considerarse. Se ha elegido, sin embargo, por ser visual y fácil de describir.

2.1.2. Objetivos relativos al diseño del agente

Los siguientes son los objetivos relacionados con el diseño del agente.

Enfoque emocional

El mecanismo de motivación y atención en aRTEA debe estar basado en un proceso emocional, ya que es dicho aspecto clave el que se ha deseado investigar en este proyecto.

Procesos operativos intrínsecos y procesos de aplicación no-intrínsecos

aRTEA debe separar explícitamente los procesos operativos, que denominaremos *Intrínsecos*, de los procesos de aplicación, que denominaremos *No-Intrínsecos*.

Toda implementación debe ofrecer un núcleo de procesos operativos pre-implementado, en el que el mecanismo de motivación y atención emocional siga la especificación aRTEA.

La funcionalidad genérica de estos procesos operativos pre-implementados quedará separada de la funcionalidad específica de los procesos de aplicación, lo que permitirá descargar a bRTEA de la necesidad de tener que desarrollar el mecanismo de motivación y atención para cada nuevo comportamiento de la aplicación, y no tener que decidir sobre cómo integrarlo con el proceso de motivación y atención de otros comportamientos en el sistema, los cuales compiten por los mismos recursos de procesamiento. Debido a esta separación explícita entre los procesos operativos y de aplicación, la descripción de los comportamientos debería ser más sencilla y clara, al estar ambas funciones desacopladas, con lo que el mantenimiento del sistema debería resultar más sencillo.

Funcionalidad del agente

aRTEA debe permitir construir agentes con un grado de autonomía suficiente para:

- Percibir su entorno y manipularlo según la aplicación para el que se ha concebido.
- Seleccionar sus problemas objetivo, a partir de objetivos generales establecidos por el usuario. Definir sus objetivos utilizando un modelo de satisfacción sobre el espacio de situaciones, de forma que permita al agente aceptar soluciones parciales al problema, más o menos satisfactorias.
- Gestionar sus recursos de procesamiento, dedicándolos a los problemas más prometedores, teniendo en cuenta las expectativas de poder resolverlos y la recompensa esperada.
- Adquirir y aplicar habilidades, tanto de resolución de problemas, como habilidades emocionales, que le permitan adaptar su actitud frente a las situaciones y mejorar su desempeño.

2.1.3. Objetivos relativos a la implementación del agente

La especificación aRTEA debe ser independiente de la tecnología de procesamiento utilizada para su implementación.

En ese sentido, aRTEA no debe restringir innecesariamente las alternativas de implementación, aunque sí debe plantear las cuestiones computacionales que deben considerarse para alcanzar los requerimientos funcionales establecidos para el RTEA. De este modo, aRTEA debe considerar el efecto de la representación del conocimiento y de la organización del comportamiento en el coste computacional, que tendrá mayor o menor impacto dependiendo del diseño del procesador elegido en cada implementación concreta.

2.2. El paradigma RTEA

aRTEA es una arquitectura de agentes artificiales, y como tal, define una forma particular de organizar los procesos del RTEA. Ese enfoque de organización, al que denominamos *Paradigma* RTEA, resulta determinante en:

1. La funcionalidad alcanzable en el RTEA una vez en explotación, y que resulta de gran interés para uRTEA, el usuario del agente.
2. El proceso de diseño y construcción del RTEA, que interesa en especial a bRTEA, el constructor del agente.

Para la especificación de aRTEA se ha seguido la siguiente secuencia de actividades:

1. Consideración de la problemática de la construcción de agentes desde un punto de vista práctico de ingeniería.
2. Revisión de los modelos de representación del conocimiento y organización del comportamiento propuestos anteriormente por otros autores, y que han sido desarrollados en la forma de arquitecturas de agentes.
3. Selección y combinación de modelos de representación del conocimiento y organización del comportamiento más apropiada y su incorporación a la arquitectura aRTEA, para cumplir los objetivos de la especificación.
4. Consideración de la factibilidad de implementación de la arquitectura propuesta con la tecnología de procesamiento disponible, evaluando los aspectos computacionales.

Se presenta a continuación las claves de la arquitectura.

2.2.1. Naturaleza de los agentes

Un RTEA es una entidad con dos componentes básicos:

1. Sistema Mental.
2. Sistema de Relación.

La Fig. 2.1 muestra el RTEA como parte de un entorno, con el que se relaciona a través del *Sistema de Relación*.

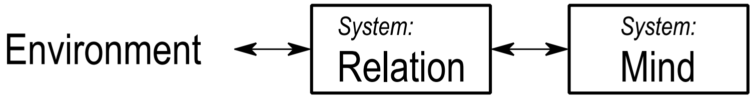

Fig. 2.1 RTEA como Sistema Mental y Sistema de Relación

Entorno

El entorno es la parte mundo real donde el RTEA es implantado y donde desarrolla su actividad, tratando de resolver los problemas que se le encomiendan.

Normalmente el mundo real, donde se resuelven los problemas de aplicación, es una parte del mundo físico, pero también puede tratarse de un entorno virtual en el que se desee resolver problemas con la ayuda del agente.

Mente

Si el entorno constituye el mundo real, en la mente del agente hay una representación lógica de dicho mundo. La mente del RTEA está constituida por el conjunto de procesos cognitivos que se desarrollan sobre dicha representación lógica.

Sistema de relación

El sistema de relación es la interfaz entre el mundo real y el mundo lógico abstracto en la mente del agente, y establece un camino bidireccional de intercambio de información; camino de entrada para las percepciones y de salida para las acciones motoras.

Soporte de ejecución

Los procesos que tienen lugar en ambos sistemas, mente y relación, son procesos de tratamiento de información que necesitan de un soporte de ejecución, que en última instancia será un procesador en el mundo físico.

Naturaleza del agente en función de su aplicación

En función de la aplicación para la que fuera concebido, un RTEA podría ser de distinta naturaleza.

Podría ser un agente físico, o parte de un sistema físico en el que estuviese embebido, como por ejemplo en el robot móvil de servicio multifuncional considerado como caso de aplicación en este proyecto.

También podría ser un agente lógico, como por ejemplo, un software de aplicación, un organizador personal o el sistema operativo de un computador.

También podría ser un componente de otro agente artificial.

2.2.2. Procesos cognitivos y comportamiento

El comportamiento de RTEA es el resultado de los procesos cognitivos que tienen lugar en su mente. Dichos procesos se organizan en dos capas de procesamiento principales:

1. Capa de Procesos de Aplicación, que produce el comportamiento efectivo del agente.
2. Capa de Procesos Operativos, encargada de la organización de los procesos de aplicación.

2.2.3. Voluntad del agente

aRTEA define de forma explícita los deseos del agente, los cuales constituyen su voluntad y son el motor de su comportamiento.

Así, el comportamiento exteriorizado del agente puede entenderse como el resultado de procesos mentales que se organizan de forma ordenada, con una intención, y no como resultado de procesos casuales.

Los otros agentes en el entorno podrán atribuir dicho comportamiento exteriorizado a la voluntad del RTEA.

La voluntad del usuario – el problema de aplicación

RTEA se construye para una aplicación de interés para uRTEA. RTEA representa a uRTEA, con el que mantiene un contrato de agencia, por lo que la voluntad del usuario está también codificada, y fusionada, con la del agente.

2.2.4. Recursos de procesamiento del agente

Los procesos mentales del agente necesitan de un soporte de ejecución, que denominamos *Procesador*.

Procesador

La naturaleza del procesador, su estructura y la tecnología que lo implementa, no es parte de la especificación de aRTEA. Sin embargo, se trata de un aspecto esencial de la construcción de los agentes, y por tanto se considera en este proyecto mediante la consideración de diferentes alternativas de implementación.

Un ejemplo de procesador mental es el *Coprocesador Emocional*, encargado de apoyar al *Procesador Principal*, procesando de forma permanente y periódica las emociones del agente. Otros ejemplos de coprocesadores son los que forman parte del *Sistema de Relación*, y que se encargan de procesar la información que llega desde los sentidos o se envía hacia los efectores. Se trata de procesadores de información. Para que dichos procesadores sean funcionales, deben poseer una estructura apropiada. Para que sean efectivos, necesitan a su vez, que concurran unas condiciones favorables, como disponer de energía, y estar apropiadamente conectados con el resto de coprocesadores del sistema cognitivo, de forma que exista una coincidencia espacio-temporal de los distintos conceptos que están siendo procesados.

Recursos de procesamiento en el entorno

Adicionalmente a los procesadores encargados de los procesos cognitivos del agente, para la resolución de los problemas de aplicación se necesitan otros recursos de procesamiento que forman parte del entorno. Dichos recursos pueden ser de distinta naturaleza, como la energía que mueve una máquina, un objeto físico a ser transformado, o un espacio libre de obstáculos donde un robot pueda realizar su trabajo.

Gestor de recursos de procesamiento

RTEA puede entenderse como una unidad de gestión de recursos de procesamiento, recursos sobre los que organiza y expresa su comportamiento.

La arquitectura aRTEA se centra sólo en los recursos de procesamiento que permiten la ejecución de los procesos cognitivos del agente, incluidos los procesos de relación.

En el nivel de la aplicación, los recursos de procesamiento son los elementos necesarios para que los procesos del mundo real se produzcan. Estos elementos se representan como conceptos en la mente del RTEA. Las habilidades del agente respecto al problema concreto de aplicación, le permiten tomar decisiones sobre cómo utilizar dichos recursos para intentar satisfacer los objetivos.

Se pueden considerar tres escenarios relacionados con los recursos y su uso:

Escenario 1 – Recursos > Problemas

Escenario 2 – Recursos = Problemas

Escenario 3 – Recursos < Problemas

Los RTEAs están concebidos para el tercer tipo de escenario, en el que la carga de trabajo supera a los recursos de procesamiento disponibles; es decir, en la agenda del agente siempre hay problemas pendientes por resolver. Este enfoque permite una explotación óptima del agente. Para ello, el *Sistema de Atención* - encargado de la organización del comportamiento del agente - aplica estrategias que intentan hacer un uso eficiente de sus recursos.

En un escenario de tipo 1, el agente no está siendo explotado de forma óptima, y si se detecta este tipo de escenario, lo normal es que se reasigne la carga de trabajo para corregir la situación. El escenario de tipo 2 sería la situación óptima, aunque difícil de alcanzar y mantener. En general, convendrá tener una carga de trabajo según el escenario 3, aunque próxima al punto de equilibrio del escenario 2.

2.2.5. RTEA como individuo

aRTEA es una especificación de agente artificial centrada en el individuo.

Un individuo tiene control sobre la utilización de sus propios recursos, aplicando su voluntad, independientemente de que dicha voluntad haya sido influenciada, condicionada, o incluso manipulada por otros. En el caso de RTEA, su voluntad está condicionada por la de su usuario. Pero dada dicha voluntad, independientemente de cómo se haya generado, el agente aplicará sus recursos para satisfacerla.

Grupos

Los grupos también tienen voluntad propia, entendida como los objetivos fundacionales de la organización. A su vez, disponen de recursos comunes. Sin embargo, en la naturaleza de todo grupo, están los individuos que lo constituyen y por tanto están también sus voluntades individuales, por lo que para que los recursos del grupo se pongan al servicio de la voluntad común, se necesita un acuerdo entre los individuos, que establezca cuáles son las prioridades del grupo frente a las individuales, y esto requiere de una capa comportamental adicional en cada uno de los individuos.

Comportamiento social del RTEA

Los RTEAs pueden presentar habilidades sociales de interrelación con otros agentes, de forma que un RTEA podría colaborar en la resolución de problemas como parte de un grupo, siguiendo el paradigma de sistemas multiagente.

Sin embargo, para aRTEA las habilidades de interrelación social, al igual que cualquier otra habilidad de resolución de problemas, forman parte de la capa de procesos de aplicación, capa considerada como no intrínseca, ya que puede implementarse o no, dependiendo de si el RTEA debe colaborar o no con otros agentes.

2.2.6. Representación del conocimiento en RTEA

El agente necesita conocer el mundo para poder comportarse en él de una forma motivada. La representación del conocimiento elegida afecta a qué conocimiento puede concebirse y también a la forma de procesar mentalmente dicho conocimiento para provocar un comportamiento.

Las siguientes son las claves de la representación del conocimiento en RTEA.

Objetos y conceptos

Un objeto es una parte del mundo real sobre la que RTEA tiene algún interés. Según el caso, RTEA puede percibir, conocer, desear y/o manipular objetos.

RTEA representa los objetos de forma abstracta, mediante conceptos (ideas) en su mente, que forman parte de su *Sistema de Creencia*.

Consciencia en RTEA

Los conceptos mantenidos por el *Sistema de Creencia* están disponibles, de forma explícita, para los procesos operativos del agente que constituyen el mecanismo de motivación y atención emocional.

Debido a esto, decimos que RTEA es consciente de los conceptos (ideas). Frente a estos, el resto de datos que no intervienen de forma explícita en los procesos de motivación y atención emocional, no se consideran conscientes.

Obviamente, el uso que se hace del término *Consciencia* en RTEA, no tiene nada que ver con el complejo fenómeno que los agentes naturales conscientes experimentan.

Situación, tiempo y cambio

RTEA representa su conocimiento en base a dos tipos de conceptos básicos:

1. Situación
2. Cambio

Una situación es la descripción del estado de un objeto en un momento dado. En RTEA el tiempo es una magnitud abstracta monodimensional utilizada para relacionar las situaciones entre sí y en base a modelos de transformación. El agente puede consultar un reloj como patrón de dicho parámetro.

El cambio es la descripción del proceso de transformación de una situación en otra (ver Fig. 2.2).

Fig. 2.2 Proceso de cambio de situación

2.2.7. Organización del comportamiento en RTEA

La organización del comportamiento afecta directamente a cómo el agente puede actuar, y ello determina sus capacidades de resolución de problemas, limitando o potenciando su funcionalidad y efectividad. También afecta a cómo se construye el agente, ya que bRTEA debe preparar los comportamientos siguiendo el método organizativo propuesto.

Modelos de comportamiento

La respuesta del agente ante la situación percibida es producida mediante la aplicación de un conjunto de habilidades que el agente tiene para tratar los problemas que surgen en el entorno. Las habilidades se representan mediante *Modelos de Comportamiento*. Estos modelos de comportamiento son específicos para cada tipo de problema, y, o bien bRTEA los incorpora durante la construcción del agente, o bien RTEA los adquiere y mejora durante su vida, como resultado de procesos de aprendizaje.

El comportamiento de RTEA puede clasificarse en dos categorías:

1. Comportamiento Exteriorizado
2. Comportamiento Interiorizado

Comportamiento exteriorizado

El *Comportamiento Exteriorizado*, o *Efectivo*, es el que puede observarse desde el entorno. RTEA influye sobre su entorno al realizar acciones motoras a través de sus efectores. Las acciones de manipulación de los objetos del mundo real cambian su situación, y otros agentes inteligentes pueden atribuir dicho cambio a RTEA y a su voluntad.

Comportamiento interiorizado

Son muchas las acciones que RTEA realiza y que no traspasan directamente la interfaz de relación. El *Comportamiento Interiorizado*, produce cambios en la situación mental de RTEA, modificando sus creencias. Los procesos de planificación, de toma de decisiones, o los procesos de aprendizaje, son ejemplos de procesos que producen dicho comportamiento interiorizado, que, aunque no afecta al entorno de forma directa, sí lo hace de forma indirecta, al modificar la actitud del agente que influirá en su comportamiento exteriorizado posterior.

2.2.8. Flujo general de información en aRTEA

La Fig. 2.3 muestra un diagrama de flujo de información en aRTEA. En la figura, los rectángulos representan procesos, las elipses conceptos, y las flechas el camino o vía que sigue el procesamiento. Los *Sentidos* y los *Efectores* son los dispositivos de interfaz con el entorno y son representados con rectángulos de trazo grueso.

Se trata de una representación de los tipos de procesos que desde un punto de vista genérico tienen lugar en el agente, con indicación del origen y destino de la información.

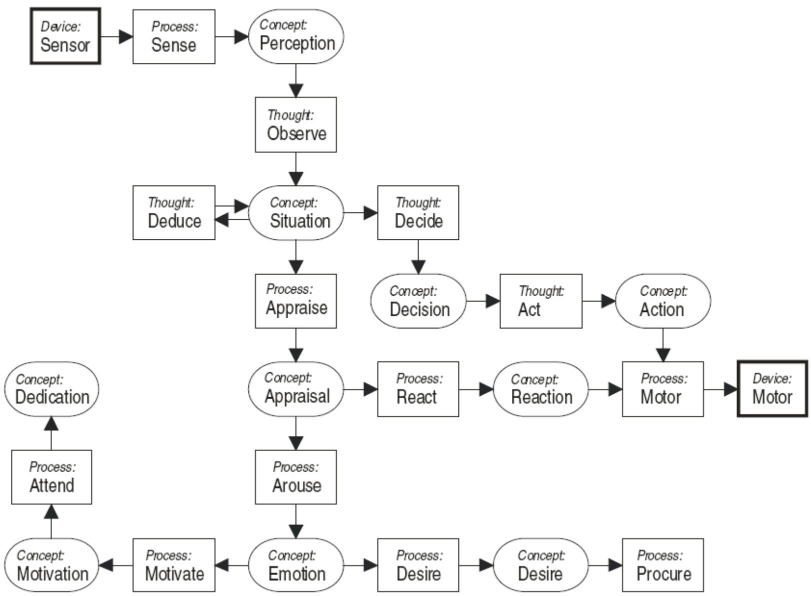

Fig. 2.3 Flujo de procesamiento general en aRTEA

El flujo de información se produce a través de las siguientes vías principales:

- Vía Sensitiva
- Vía Motora
- Vía Deliberativa
- Vía Reactiva
- Vía Emocional
    - Sub-vía de Deseo
    - Sub-vía de Motivación y Atención

Los procesos que producen el comportamiento efectivo, fluyen desde los sentidos, con la percepción, hacia los efectores, con la acción motora (ver Fig. 2.4 - camino a). El agente percibe la situación actual de su entorno real a través de la *Vía Sensitiva* y manipula el entorno, influenciando su evolución, a través de la *Vía Motora*.

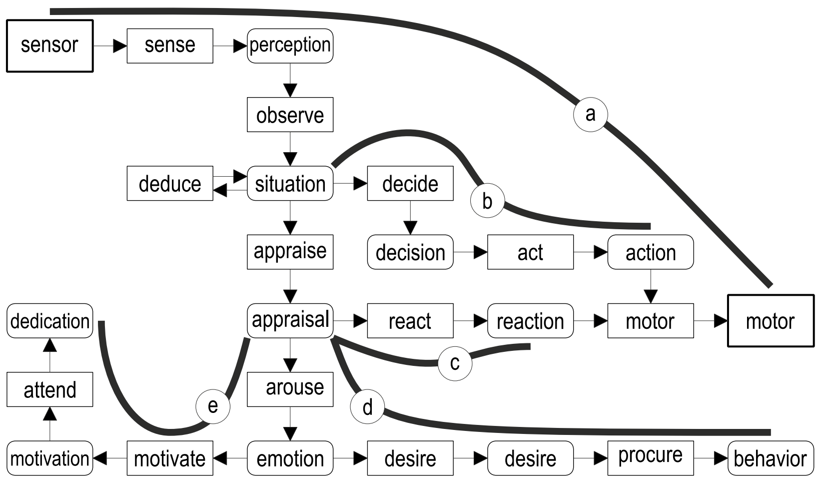

Fig. 2.4 Vías de procesamiento

Los procesos que producen el comportamiento interiorizado, y que contribuyen posteriormente al comportamiento exteriorizado, fluyen a través de dos grupos de caminos: los que tienen que ver con los procesos de aplicación (resolución de problemas), y los que tienen que ver con los procesos operativos (motivación y atención).

Siguiendo las vías *Deliberativa* (Fig. 2.4 b) y *Reactiva* (Fig. 2.4 c) se procesan, o bien de forma deliberativa, o bien de forma reactiva, distintos conceptos relacionados con los problemas de aplicación. A través de estas vías, RTEA realiza observaciones, deducciones, apreciaciones, decisiones, acciones deliberadas y reacciones.

Siguiendo las vías emocionales: *Vía de Deseo* (Fig. 2.4 d) y *Vía de Motivación y Atención* (Fig. 2.4 e) se procesan las emociones. A través de estas vías, RTEA genera nuevos deseos, y motiva y atiende los pensamientos de procuración de dichos deseos.

Reacciones y deliberaciones

Según la estructura interna de los procesos de comportamiento, RTEA distingue entre:

- Comportamiento Reactivo
- Comportamiento Deliberativo

Esta distinción es importante, ya que ambos tipos de comportamiento están integrados de forma diferente en el mecanismo de motivación y atención emocional de RTEA.

Vía reactiva – reacciones

La *Vía Reactiva* conecta directamente los sentidos con los efectores a través de procesos reactivos, los cuales producen el *Comportamiento Reactivo* del RTEA. Los procesos reactivos tienen tiempos de respuesta acotados que se pueden garantizar. Normalmente el tiempo de respuesta es breve, si se compara con los tiempos de respuesta de los procesos deliberativos.

Las *Reacciones* son necesarias para responder ante ciertas situaciones del entorno dentro de un plazo de tiempo acotado. Dicho plazo está relacionado con la dinámica del problema. RTEA no puede definir reacciones para todas las situaciones (estímulos) que se puedan percibir, pero sí para las más importantes, por lo menos desde el punto de vista de la supervivencia del agente, o de las prestaciones mínimas que la implementación debe asegurar frente a un problema de aplicación.

La selección de las reacciones a implementar es una cuestión de diseño importante que bRTEA debe abordar.

El tiempo de respuesta del *Comportamiento Reactivo* se puede garantizar sólo si no hay interdependencias complejas entre los conceptos intermediarios en los procesos involucrados, de forma que no haya incertidumbres sobre la disponibilidad de los conceptos. No debería permitirse situaciones potenciales de bloqueo o espera indefinida, ni encadenamiento de un número de procesos discretos no determinista.

Las reacciones suelen pues obedecer a modelos de comportamiento (métodos) estructuralmente simples, aunque puedan involucrar extensos conjuntos de conceptos.

En cualquier caso el tiempo de proceso se debe pre-acotar, con lo que si el volumen de información procesada en la reacción es elevado, pueden requeriste procesadores de altas prestaciones, para poder garantizar los tiempos de respuesta preestablecidos.

Reacciones y consciencia

RTEA no es consciente de una reacción en el momento en que se produce. Esto es debido a que los conceptos intermediarios en la reacción no intervienen en el mecanismo de motivación y atención emocional. Para dicho mecanismo, una reacción es un proceso atómico.

Sin embargo, el propio proceso de reacción puede ser percibido de forma consciente, así como la causa (estímulo) y el efecto (respuesta). Esta conceptualización de la reacción, si se produce, lo hace como un proceso separado de la reacción que se conceptualiza, y su tiempo de respuesta puede exceder al de la propia reacción, con lo que se suele ser consciente de la reacción a posteriori.

Vía deliberativa – deliberaciones

La *Vía Deliberativa* conecta los *Sentidos* con los *Efectores* a través de procesos deliberativos, los cuales producen el *Comportamiento Deliberativo* del RTEA. Un proceso deliberativo, visto por el proceso de motivación y atención emocional, tiene una estructura con conceptos intermediarios, con lo que no es siempre posible acotar el tiempo de respuesta. En general, el tiempo de respuesta de las deliberaciones suele ser mayor que el de las reacciones.

Muchos de los procesos de resolución de problemas de aplicación son deliberativos. Una planificación o una decisión, pueden requerir de un procesamiento iterativo, o de la selección de alternativas que dependan de conceptos intermediarios.

Deliberaciones y consciencia

Las deliberaciones están integradas en el sistema consciente del agente, ya que los conceptos intermediarios de las deliberaciones forman parte del conjunto de creencias, e intervienen en el proceso de motivación y atención emocional.

Vía emocional – emociones

La organización de la ejecución de los procesos de aplicación - la selección de los problemas a tratar y qué alternativas de solución considerar y aplicar, es dirigida por el proceso de atención. La atención es un proceso operativo que intenta optimizar el desempeño de los procesos de aplicación.

En RTEA el proceso de atención es un proceso dirigido emocionalmente.

Los procesos operativos de motivación y atención emocional, son procesos reactivos con tiempo de respuesta breve y garantizado. Para garantizar dicho tiempo de respuesta, si el número de problemas manejados por el agente es elevado, es necesario un procesador de altas prestaciones. Esta tesis plantea distintas alternativas para acelerar dicho proceso.

La *Vía Emocional*, tiene dos ramas principales: la *Vía del Deseo* y la *Vía de la Motivación y Atención*.

Sobre la *Vía del Deseo*, como parte del proceso de resolución de problemas, se formulan y mantienen nuevos *Deseos* (situaciones objetivo), lo que provoca la aparición de nuevos problemas debido a las discrepancias entre las situaciones deseadas (hipotéticas) y las situaciones actuales (reales), y como respuesta en RTEA, la construcción automática de nuevos procesos, denominados *Pensamientos*, para la procuración de los deseos.

Sobre la *Vía de la Motivación y Atención*, la situación del propio proceso de resolución de problemas es apreciada emocionalmente, de forma que el agente considera cuestiones relevantes como la importancia del problema a resolver y su expectativa de resolución.

Como resultado de esta apreciación, se establece un nivel de motivación para el pensamiento que procura por el problema, lo que contribuye a dirigir la atención del agente hacia los pensamientos potencialmente más productivos.

2.3. Diseño estructural

Se presenta a continuación el diseño estructural de RTEA.

2.3.1. Descomposición modular del sistema cognitivo

RTEA se descompone en 5 sistemas principales. Esta descomposición facilita la descripción de la arquitectura y su implementación.

Para realizar esta descomposición modular se ha aplicado las recomendaciones de la comunidad científico-técnica que durante más de 70 años ha sentado las bases de la ingeniería del software. En ese sentido, se ha intentado desacoplar en la medida de lo posible cada uno de los módulos, manteniendo su cohesión con el resto del sistema. Para ello se ha definido los módulos basándose fundamentalmente en criterios funcionales, lo que facilita la escalabilidad del sistema.

Los 5 sistemas en RTEA son:

- Sistema de Creencia
- Sistema de Comportamiento
- Sistema de Emoción
- Sistema de Atención
- Sistema de Relación

La Fig. 2.5 muestra dicha descomposición, con el *Sistema de Relación* actuando como interfaz entre el entorno y los otros 4 módulos en los que se descompone la mente del agente.

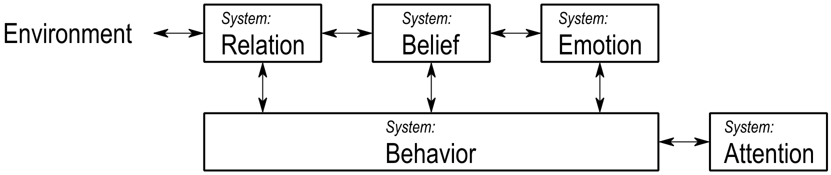

Fig. 2.5 Descomposición modular de RTEA

Las interfaces de cada uno de los módulos, representadas de forma simplificada mediante flechas en la Fig. 2.5, permiten relacionar las distintas funciones del agente.

Con objeto de minimizar el número de interfaces y así reducir la complejidad, se le ha dado al *Sistema de Creencia* el papel de módulo vertebrador. Es en dicho módulo donde el agente mantiene una imagen interna del entorno y del problema de aplicación.

A continuación se presenta las funciones principales de cada uno de los 5 módulos y la justificación para dicha descomposición.

2.3.2. Sistema de creencia

El *Sistema de Creencia*, Fig. 2.6, mantiene el conocimiento que el agente tiene sobre sí mismo y sobre su entorno. Basándose en dicho conocimiento, el agente puede comportarse.

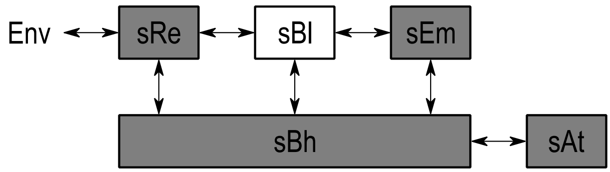

Fig. 2.6 Sistema de creencia

Realidad - mundo real

La existencia de un mundo real, de los objetos en dicho mundo, y de sus relaciones, es tratada por la metafísica, y en especial por la ontología, que se pregunta sobre qué cosas existen y cómo deberíamos clasificarlas.

Un área donde la ontología se ha desarrollado de forma práctica ha sido la informática, en especial en la inteligencia artificial, donde han tenido que desarrollarse sistemas de representación del conocimiento que soportasen procesos cognitivos como la percepción y la toma de decisiones.

RTEA supone la existencia de un mundo real. Es el mundo donde surgen sus problemas y donde desarrolla su comportamiento. Al mundo real se accede a través del módulo de relación, observándolo con los sentidos y manipulándolo con los efectores. Del mundo real se tiene una imagen mental, para la que se necesita una representación apropiada.

En muchas aplicaciones de agentes enfocadas al control de sistemas, el mundo real es el mundo físico. Un ejemplo es la aplicación de control de la plataforma de robot móvil de servicio considerada como caso de estudio en este proyecto. Sin embargo, el mundo real del agente también puede ser un mundo virtual o lógico, como por ejemplo en una aplicación para el apoyo en la demostración de teoremas matemáticos.

Representaciones del conocimiento

Una representación del conocimiento debe permitir realizar inferencias, mediante procesos de razonamiento, de forma que a partir del conocimiento disponible se genere conocimiento nuevo, o que se explicite el conocimiento que ya se tenía de forma implícita.

Se han definido “ontologías” con distintas pretensiones. Las más comunes son ontologías definidas ad hoc en un contexto de aplicación limitado. Un debate intenso y todavía abierto se centra en si es posible definir una ontología de aplicación general.

Representación del conocimiento en aRTEA

El mundo real de RTEA abarca todos los objetos que son concebibles por el agente. Aunque es fundamental la representación del conocimiento que se elija, aRTEA deja parcialmente abierta la elección que bRTEA puede realizar sobre la misma.

Esto es debido a que aRTEA especifica únicamente los procesos operativos de motivación y atención emocional, y a que los procesos de aplicación son como cajas negras para dicho mecanismo.

La representación del conocimiento en los procesos de aplicación será una decisión de diseño de bRTEA. Dicha decisión estará influenciada por el problema de aplicación, y por as técnicas que bRTEA desee implementar para resolverlo. Aun así, los procesos de aplicación deben implementar la interfaz que requiere el sistema de motivación emocional y atención para poder organizar su ejecución, con lo que en la representación del conocimiento elegida se deben poder representar de forma explícita las variables involucradas en dicha interfaz.

Conceptos (ideas)

aRTEA representa el mundo real mediante conceptos. Sobre estos conceptos tienen lugar los procesos mentales.

Cualquier pieza de conocimiento que se genera, transforma y utiliza, si se define de forma explícita para el mecanismo de motivación y atención, entonces es un concepto. Decimos que RTEA es consciente de los conceptos.

Al ser los conceptos representaciones personales que RTEA elabora de la supuesta realidad, decimos que son creencias del RTEA.

Ciclo de vida de las creencias

Los conceptos (ideas) tienen un ciclo de vida, que se refiere a su producción, actualización, consumo y destrucción.

Las situaciones en particular, y los conceptos en general, tienen un “lugar” en el esquema de conocimiento del agente, organizado de forma jerárquica en una estructura en árbol de contextos de creencias. Esta estructura se corresponde con el árbol de problemas-subproblemas. De ese modo, cada creencia forma parte de un contexto, que también tiene su propio ciclo de vida, con lo que el ciclo de vida de una creencia está circunscrito en el de su contexto.

Así, las creencias construyen su soporte de memoria cuando se construye el contexto de problema donde tienen utilidad. En ese contexto hay procesos que necesitan de las creencias para poder ejecutarse, son los procesos consumidores de conocimiento.

Las creencias se producen y toman un valor cuando los procesos productores se ejecutan. La ejecución de dichos procesos en el contexto de problema está dirigida por el sistema de atención, considerando la motivación emocional de cada uno de dichos procesos.

Durante su ciclo de vida, y de forma general, las creencias pueden producirse (construirse y/o actualizarse) múltiples veces. Como casos límite, podrían producirse sólo 1 vez, o no producirse nunca, si el contexto de problema es destruido antes de la ejecución de la producción del conocimiento.

La duración temporal de una creencia se produce desde el punto temporal de la creación de su contexto de problema hasta la duración de la vida de éste, que en el límite podría alargarse hasta la duración de la vida del agente.

Producción de los conceptos en aRTEA

Las creencias son producidas por procesos mentales, e incorporadas a un sistema de memoria que permite compartirlas entre los procesos mentales productores y consumidores (Fig. 2.7).

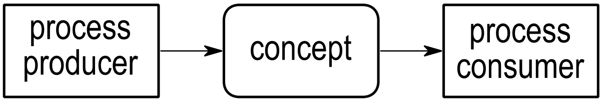

Fig. 2.7 Producción y consumo de conceptos

Los procesos productores aplican los siguientes métodos:

- Método procedural
    - Toma de una decisión
        - Generación de un plan
        - Decisión de una acción
    - Interpretación de una situación
        - Observación - adaptación de la situación para ser procesada de forma más conveniente en un contexto de problema dado.
        - Apreciación – valoración subjetiva de la situación, para tomar decisiones con un propósito dado.
    - Justificación o explicación de una situación pasada o actual, utilizando un modelo dinámico explicativo.
    - Previsión de una situación futura, utilizando un modelo dinámico predictivo.
    - Deseo de una situación futura, como parte de un método de resolución de problemas.
- Método creativo - generación aleatoria de deseos y su valoración, la cual puede conducir a descartar el deseo generado, o a aceptarlo, siguiendo el mecanismo de flashes de lucidez.

El *Sistema de Creencia* permite compartir el mismo conocimiento entre varios procesos consumidores como muestra la Fig. 2.8.

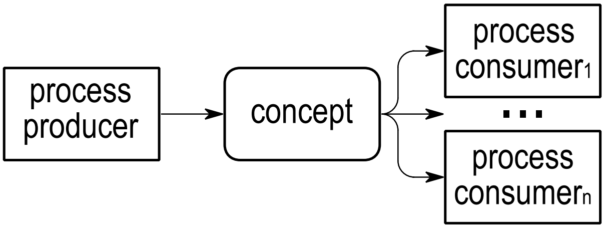

Fig. 2.8 Compartición del conocimiento

El *Sistema de Creencia* permite a su vez, gestionar las distintas versiones producidas de un mismo concepto, mediante un proceso de fusión de conocimiento (Fig. 2.9).

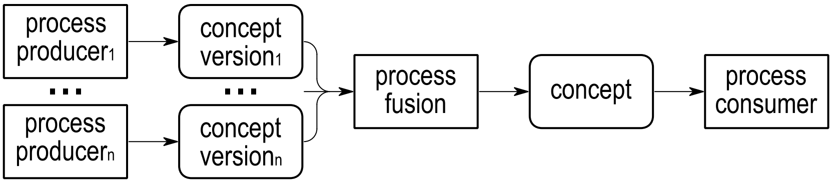

Fig. 2.9 Fusión de versiones

Mantenimiento de los conceptos en aRTEA

Las creencias son mantenidas por procesos de memoria (Fig. 2.10), los cuales pueden ser de los siguientes tipos:

- Memorización de un concepto de forma temporal he incorporación al conjunto de creencias
    - Sin procesado previo
    - Con procesado previo (poda/simplificación, recodificación, combinación con otros conceptos)
- Revisión de un concepto memorizado (refresco)
    - Actualización
        - De su valor
            - Debido a una actualización previa de su método de producción
            - Debido a una actualización de los conceptos que lo soportan
        - De su confianza
            - Mediante un modelo temporal simplificado
            - Mediante un modelo de relación general
    - Restructuración de la representación
        - Con un incremento de las características representadas
        - Con una reducción de las características representadas
- Olvido de un concepto, cuando se destruye definitivamente su contexto de creencias al perder su utilidad
- Recuerdo de un concepto previamente memorizado, recuperándose de la memoria mediante un proceso de recuperación asociativo

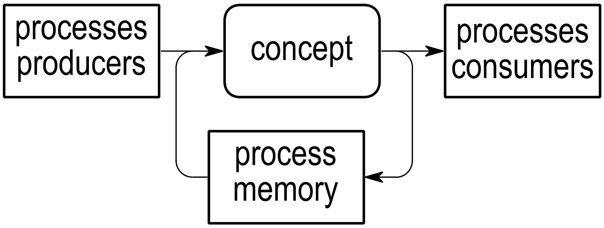

Fig. 2.10 Proceso de memoria

Representación de la situación y del cambio

aRTEA utiliza una representación situacional del mundo real. En esta representación el conocimiento es organizado sobre la línea temporal (tiempo cartesiano y absoluto).

Los procesos de cambio o de transformación de situaciones relacionan entre sí las distintas situaciones sobre la línea temporal.

La situación – el instante

Una *Situación* es un concepto que describe una parte del mundo real en un instante de tiempo dado.

Un *Instante* se refiere realmente a un intervalo de tiempo entre dos puntos temporales. Los puntos temporales no pueden descomponerse más sobre la línea de tiempo. La duración de un instante puede medirse de forma objetiva con un reloj, sin embargo RTEA realiza apreciaciones subjetivas sobre las duraciones, las cuales están influenciadas por el contexto en el que se realiza la medida y por su utilidad.

Características de la situación

La situación, o mejor dicho, la descripción de la situación (como concepto), tiene las siguientes características:

- Se refiere sólo a una parte del mundo real: el objeto descrito.
- Se produce mediante un proceso de abstracción, pues considera sólo los aspectos del objeto que son relevantes en el contexto de problema en que la descripción de la situación se va a utilizar.
- Es aproximada, porque no es completa, y siempre se podría detallar otros aspectos adicionales del objeto descrito.
- Es imprecisa, porque las fuentes de información utilizadas para producirla no ofrecen una confianza total.
- En resumen, es subjetiva (o personal), porque se produce para un propósito dado, en el contexto de un problema. Con lo que: el objeto sobre el que se enfoca, la selección de las facetas que se consideran relevantes, el grado de detalle con el que se describen, y las fuentes de información consultadas, están condicionadas por la motivación que se tiene en el problema.

El cambio

El cambio es producido por un proceso de transformación de la situación, que es conceptualizado mediante un *Proceso* (Fig. 2.11).

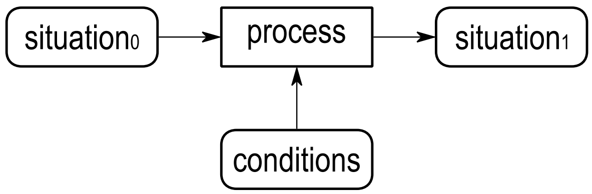

Fig. 2.11 Proceso de cambio de situación

Este proceso puede ser caracterizado mediante las siguientes propiedades:

- Situación Inicial (s_0) e Instante Inicial (t_0) – Situación en el punto temporal en el que se inicia la transformación.
- Situación Final (s_1) e Instante Final (t_1) – Situación en el punto temporal en el que finaliza la transformación.
- Sincronización de Entrada – Proceso de interfaz en el que se establece las condiciones de inicio de la transformación.
- Sincronización de Salida – Proceso de interfaz por el que se observa el resultado del proceso de transformación.
- Tiempo de Proceso – Diferencia temporal entre el instante inicial y final.
- Condiciones – Las condiciones que permiten la transformación. Cuando las condiciones se dan, el cambio se produce de forma inevitable.

El comportamiento y el cambio

El agente favorece procesos de cambio al establecer las condiciones que los favorecen. El comportamiento del agente consiste en establecer dichas condiciones. Hay sin embargo condiciones no controlables (Fig. 2.12).

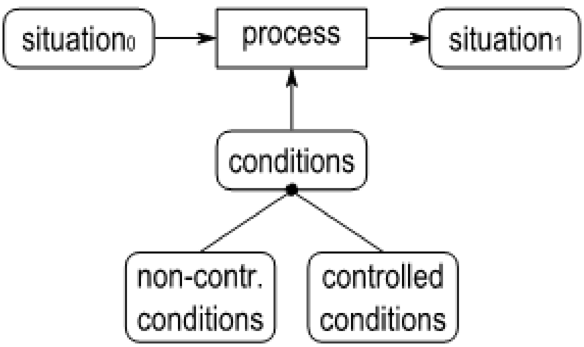

Fig. 2.12 Condiciones controlables y no controlables

Las decisiones de comportamiento del agente consisten en diseñar las condiciones controlables, dadas el resto de condiciones, de forma que la evolución de la situación sea la deseada. Para ello el agente aplica un método.

- Método (habilidad) – Una descripción de la secuencia de condiciones que deben establecerse en el tiempo para que el proceso se complete según un plan. El método es pues un plan de establecimiento de condiciones intermediarias.
- Condiciones Controlables – De las condiciones que habilitan un proceso, un subconjunto de las mismas pueden entenderse como el controlador del proceso.
- Condiciones No Controlables – Condiciones que no se pueden controlar, pero que se pueden tener en cuenta a la hora de definir el método y de ajustar las condiciones controlables.

Procesador

Para que un proceso suceda se requiere un procesador. Un procesador es un concepto definido por conveniencia. Ayuda a visualizar en qué parte del mundo se está produciendo el proceso y qué condiciones son las que se están dando para que el proceso se produzca, con lo que es fácil identificar las condiciones que llamamos controlables y que nos permiten actuar sobre el proceso, iniciándolo, encaminándolo y finalizándolo.

Valoración de la situación

Las situaciones, que se describen mediante características objetivas, son siempre acompañadas de apreciaciones subjetivas, que valoran la utilidad de la descripción de la situación y su aplicabilidad en la toma de decisiones.

Apreciación de confianza

La lista de apreciaciones que pueden asociarse a las situaciones es variable, y definida por bRTEA en función de la aplicación. Ejemplos de apreciaciones pueden ser: la objetividad, la precisión, la completitud, etc.

El mecanismo de motivación y atención emocional, sin embargo, requiere de una apreciación específica para su funcionamiento. Dicha apreciación, considerada intrínseca en el sistema, es la apreciación de *Confianza* (ver Fig. 2.13), la cual resume en un único valor la calidad de la representación que se tiene de la situación. La Confianza interviene como una contribución para la evaluación del estado emocional.

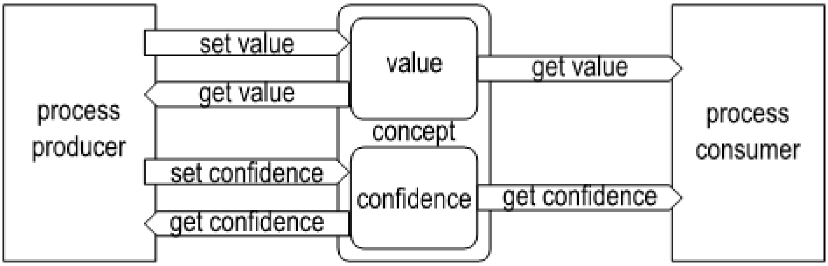

Fig. 2.13 Apreciación intrínseca de *Confianza*

Generación de la confianza

Cada actualización de una situación por parte de un proceso productor se acompaña de un refresco de su valor de confianza. Éste valor se define a partir de las confianzas de los conceptos de soporte y del método de producción aplicado.

La propagación de confianzas produce siempre, o bien un mantenimiento del valor, o una disminución del mismo. Sólo en los procesos de fusión de conocimiento, cuando hay más de una versión coincidente del concepto, la confianza puede incrementar su valor.

Actualización temporal de la confianza

El sistema de creencias actualiza las confianzas de sus conceptos aplicando un modelo temporal de degradación. Este modelo resume las circunstancias probables de disminución de la aplicabilidad del concepto a una medida temporal. Es un método de actualización simplificado que puede procesarse de forma eficiente. El proceso productor debe proveer el modelo de degradación temporal de la confianza en cada nueva producción del concepto. Por defecto, el modelo puede ser un valor constante en el tiempo.

Sincronización flexible entre producción y consumo

La apreciación de confianza permite la sincronización flexible entre los procesos productores y consumidores, ya que siempre hay una versión del concepto en la memoria. De esta forma no existen bloqueos estrictos entre los procesos productores y consumidores.

Es el proceso consumidor el que resulta más o menos motivado para utilizar el valor de la situación en función de su nivel de confianza. Una situación de baja confianza desmotiva el proceso consumidor a la vez que motiva el proceso productor. Sin embargo esto no supone un bloqueo estricto. Pudiendo tener el proceso consumidor otros factores motivantes que compensen la desmotivación causada por una baja confianza en el concepto considerado.

2.3.3. Sistema de comportamiento

El *Sistema de Comportamiento* (Fig. 2.14), implementa los procesos de comportamiento. Los procesos de transformación del entorno atribuibles al comportamiento del agente se generan a partir de procesos mentales, los cuales se traducen en comportamiento efectivo a través del *Sistema de Relación*.

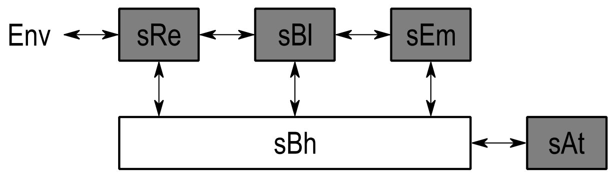

Fig. 2.14 Sistema de comportamiento

Características del comportamiento en aRTEA

Para afrontar problemas en entornos dinámicos con incertidumbre, los RTEAs solapan los procesos de decisión y acción, tomando decisiones racionalmente (mediante deliberaciones) y emocionalmente (mediante reacciones) y actuando cuando los planes están todavía incompletos. Los planes se rediseñan de forma continua.

Procesos mentales en aRTEA

Los procesos mentales en aRTEA son procesos de información. Son productores y consumidores de conocimiento. Utilizan métodos que contienen las condiciones de transformación de los procesos consumidos en producidos. Estas condiciones son las necesarias para que un procesador, que actúa como soporte, desarrolle el proceso mental. Las condiciones son definidas mediante estructuras de control de flujo, que pueden reducirse a primitivas de secuencias, selecciones y repeticiones, utilizando un lenguaje que selecciona condiciones de transformación atómicas o instrucciones.

Los procesos mentales son clasificados en procesos de utilidad y procesos operativos. Los procesos de utilidad están relacionados con el comportamiento efectivo de RTEA. Los operativos, con la organización de la ejecución de los anteriores:

- Procesos de Utilidad
  - Procesos de Percepción
  - Procesos de Observación
  - Procesos de Decisión
  - Procesos de Acción
  - Procesos de Reacción
- Procesos Operativos
  - Procesos de Emoción
  - Procesos de Motivación
  - Procesos de Atención
  - Procesos de Memoria
- Procesos de Aprendizaje
  - Procesos de Aprendizaje Emocional
  - Procesos de Aprendizaje de Habilidades de Aplicación

Motor del comportamiento – problema

El comportamiento en RTEA se centra en el concepto de problema. El motor del comportamiento en RTEA es la motivación por resolver los problemas.

Deseos y problemas

Se genera un deseo en el proceso de resolución de un problema, lo cual, en general, genera otro problema (sub-problema)

Un problema surge cuando hay una discrepancia entre la situación actual y la situación deseada.

2.3.4. Sistema de emoción

El *Sistema Emocional* (Fig. 2.15), implementa los procesos emocionales del agente.

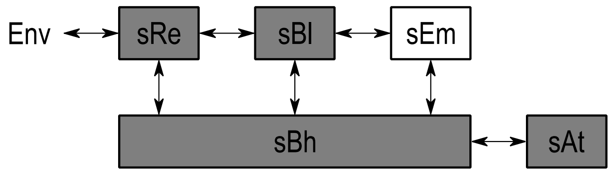

Fig. 2.15 Sistema de emoción

Se ha considerado, para un mejor diseño de la arquitectura, el separar los procesos emocionales del resto de procesos de comportamiento del agente y agruparlos en un sistema específico. Esta separación tiene ventajas: por una parte, permite identificar mejor los procesos de motivación emocional, por otra, intercambiar entre distintas implementaciones de agentes esquemas emocionales bien establecidos.

Proceso emocional

Un proceso emocional genera una respuesta ante una situación, como muestra la Fig. 2.16.

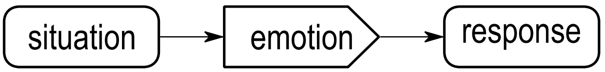

Fig. 2.16 Proceso emocional

Se trata de un proceso reactivo con un tiempo de respuesta breve. Llamamos Emoción tanto a la descripción de dicho proceso como al propio proceso.

En RTEA el proceso operativo de motivación emocional consiste en el procesamiento de múltiples emociones cada ciclo de atención. En general se trata de un conjunto extenso de emociones. Las emociones están vinculadas a los pensamientos, y responden reactivamente frente a las situaciones de los problemas tratados por los pensamientos.

Naturaleza de la respuesta emocional

La respuesta de la emoción consiste en la motivación de un comportamiento que va a gestionar la situación (ver Fig. 2.17).

Fig. 2.17 Naturaleza de la respuesta emocional

El proceso culmina con la *Motivación* de un comportamiento potencial que trataría la situación desde el punto de vista en que ésta se consideró.

La motivación no implica que el comportamiento se materialice de forma efectiva, ya que es el *Sistema de Atención* el que dirige el comportamiento del agente, utilizando los distintos niveles de motivación y actuando como un selector de alternativas.

Fases del proceso emocional

El proceso emocional ocurre en dos fases: activación y respuesta (Fig. 2.18).

Fig. 2.18 Activación, estado y respuesta emocionales

Entre la situación y la motivación del comportamiento, el proceso emocional pasa por un proceso intermediario, el Estado Emocional, que es percibido de forma consciente por el agente. Así, un proceso inicial de activación emocional se produce al apreciar la situación, provocando que el agente alcance el estado emocional, y un proceso final de respuesta emocional motivará un comportamiento en función del estado emocional alcanzado.

Fase de activación emocional

La fase de activación se inicia con la apreciación de una situación (Fig. 2.19).

Fig. 2.19 Apreciación de la situación

La apreciación es subjetiva, pues se trata de una interpretación de la situación desde un punto de vista determinado. Puede haber otros puntos de vista y por tanto otras interpretaciones. Esta interpretación está relacionada con la utilidad del proceso emocional en el contexto de problema en el que ocurre. Así, una misma situación puede producir distintas apreciaciones. A su vez, cada apreciación de situación puede contribuir a distintos estados emocionales, normalmente de forma ponderada junto con otras apreciaciones, como se muestra en la Fig. 2.20.

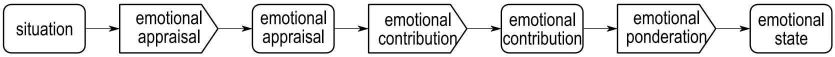

Fig. 2.20 Activación emocional

Representación de las apreciaciones

Las apreciaciones son definidas por un enunciado lingüístico sobre la situación que aprecian y un valor numérico. El enunciado está relacionado con un uso de la situación y su apreciación. El valor numérico de la apreciación se interpreta como el grado de confianza que se tiene en el enunciado lingüístico de la apreciación.

Los valores de las apreciaciones son valores normalizados en el rango [-1..+1]

- Significando -1 que se tiene desconfianza total en el enunciado, o que se tiene confianza total en que el enunciado no es válido.
- Significando +1 que se tiene confianza total en el enunciado de la apreciación.
- Significando 0 que no se tiene capacidad para afirmar un nivel de confianza en la apreciación. Se trata de una indeterminación de la apreciación.

Normalizar la representación de las apreciaciones facilita diseñar los coprocesadores encargados de evaluarlas.

Fase de respuesta emocional

Como se ha dicho previamente, la respuesta de la emoción consiste en la motivación de un comportamiento que va a gestionar la situación.

Modulación del proceso emocional

El hecho de que el *Estado Emocional* sea una situación consciente, permite al agente regular mejor el proceso emocional completo.

Actitud y carácter

La actitud se define mediante un conjunto de parámetros que afecta a los métodos de los procesos emocionales de forma general (Fig. 2.21).

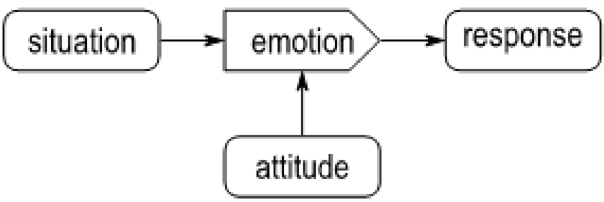

Fig. 2.21 Influencia de la actitud en el proceso emocional

La actitud es cambiante y depende del estado interno del agente que es resultado de los episodios vividos recientemente.

Efecto de la actitud en la emoción

La actitud tiene un gran efecto sobre los procesos emocionales. Afecta en ambas fases del proceso: activación y respuesta, como muestra la Fig. 2.22.

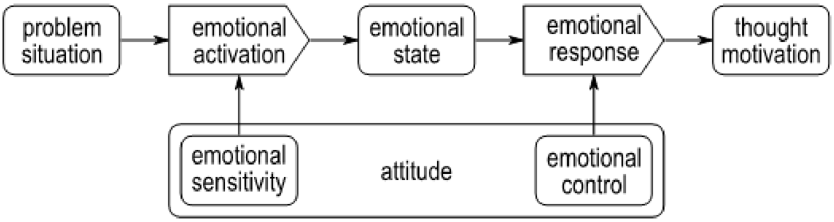

Fig. 2.22 Influencia de la actitud

Activación emocional - sensibilidad emocional

Por una parte, la activación emocional no es invariable dada una misma situación del problema, sino que depende del estado emocional general del agente y por ende de su actitud, que modifican la sensibilidad del proceso de activación.

Respuesta emocional – autocontrol emocional

Por otra parte, dado un estado emocional dado, el proceso de respuesta emocional también puede ser controlado por la actitud. Este autocontrol emocional es esencial en la organización del comportamiento de los agentes emocionales, permitiendo introducir reglas de comportamiento, muy útiles por ejemplo en ambientes sociales.

Carácter

La actitud, a su vez, se ajusta continuamente como resultado de los sucesos recientes. Dicho ajuste está modulado por otro conjunto de parámetros emocionales más estables en el tiempo, que constituye el *Carácter* del agente (Fig. 2.23).

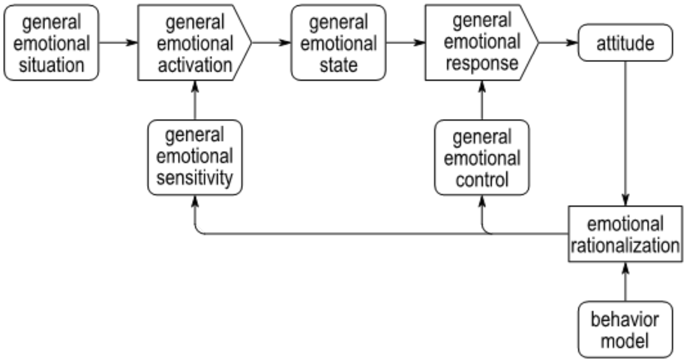

Fig. 2.23 Ajuste de la actitud

Clasificación de las emociones

Las emociones se clasifican para facilitar el modelado y la implementación. Una clasificación permite una mejor conceptualización del fenómeno a la vez que facilita diseñar los procesadores.

Dependiendo de su contribución al proceso de motivación y atención, RTEA clasifica las emociones en dos categorías básicas:

1. Emociones Intrínsecas (independientes de la aplicación)
2. Emociones No Intrínsecas (aplicación)

Las emociones no intrínsecas, dependientes de la aplicación, se clasifican en función del tipo de comportamiento que motivan. De forma genérica, aRTEA propone la siguiente clasificación:

- Emociones Positivas
  - Comportamientos de Continuación o Permanencia
  - Comportamientos de Prosecución
- Emociones Negativas
  - Comportamientos de Escape
  - Comportamientos de Evitación
- Emociones Neutras (Proactivas)
  - Comportamientos de Investigación

Utilizamos la representación mostrada en la Fig. 2.24 para representar los tres tipos de emociones no intrínsecas.

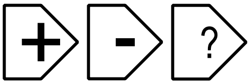

Fig. 2.24 Representación de emoción positiva, negativa y neutra

Adicionalmente, bRTEA puede subclasificar e identificar las emociones que forman parte del diseño de la aplicación.

Con objeto de facilitar la descripción del sistema, resulta conveniente etiquetar las emociones de la aplicación según la terminología utilizada en psicología. Las clasificaciones de las emociones que se encuentran en la literatura consideran *Emociones Básicas* o primarias (de 4 a 8 según los autores) y *Emociones Derivadas* o secundarias (combinaciones de emociones básicas) Por ejemplo, una clasificación de las emociones básicas muy común, considera las emociones de *Alegría*, *Tristeza*, *Miedo* e *Ira*.

Intensidad de la emoción

Una característica del proceso emocional, añadida a la valoración de positividad o negatividad del mismo, es que el estado emocional alcanzado puede ser de menor o mayor intensidad. En ese sentido hay un signo de la emoción (positiva/negativa) y una intensidad [0, +1]. En la clasificación de las emociones y su vinculación a comportamientos específicos, afecta la intensidad de la emoción.

Momento temporal de la emoción

El proceso emocional es siempre un proceso en presente. La emoción no se produce en el futuro ni en el pasado. La emoción (el proceso emocional real) no se puede simular, aunque sí se pueda simular comportamientos relacionados con estados emocionales. Sin embargo, la apreciación de una situación pasada, presente o futura puede desencadenar el proceso emocional actual en el presente.

Esto nos permite clasificar las emociones negativas/positivas en distintos subtipos según el horizonte temporal que se está apreciando, y en relación con los comportamientos que motivan. Así, pueden motivarse comportamientos de escape que se producen cuando la situación actual en la que el agente se encuentra es apreciada como negativa, mientras que también pueden motivarse comportamientos de evitación frente a situaciones que el agente no está viviendo actualmente pero que prevé puede afectarle en el futuro con lo que motiva medidas preventivas.

Emociones intrínsecas

Las emociones intrínsecas se asocian a todos los contextos de problema de forma intrínseca, independientemente del tipo de problema.

Una única emoción intrínseca se encarga de motivar un comportamiento de resolución de problema. Las apreciaciones que contribuyen a la emoción intrínseca valoran el proceso de resolución del problema desde un punto de vista general. En ese sentido, valoran si el problema es importante o no, si está bien definido o no, si se tienen o no se tienen capacidades para abordarlo, si has expectativas de éxito en su resolución o no.

Emociones no intrínsecas

Las emociones no intrínsecas se asocian a los contextos de problema de una manera que depende del propio problema y del conocimiento que se tenga en su naturaleza y en su posible solución.

Así, las emociones no intrínsecas consideran la situación del propio problema; no sobre cómo el agente podría resolverlo, sino cómo se podría resolver en general, con los recursos suficientes, basándose en la experiencia sobre la dinámica del propio problema.

Las emociones no intrínsecas detectan situaciones específicas dentro de la situación general del problema que pueden ser tratadas con métodos conocidos y se encargan de plantear situaciones sub-objetivo dentro del marco del problema general.

Son un mecanismo regulador de un proceso de resolución general de problemas ajustado a problemas específicos según la experiencia acumulada en su tratamiento. Este mecanismo general de problemas identifica secuencias de pasos, y conjunto de alternativas, y los materializa para un tipo de problemas dados. Incluso los tipos de problemas sobre los que no se tiene experiencia, desencadenan a su vez procesos de investigación.

Emociones positivas y comportamientos de continuación o permanencia

Cuando la situación actual se aprecia como favorable, una emoción positiva motiva un *Comportamiento de Continuación o Permanencia* en la situación actual. Este comportamiento consiste en realizar, activa o pasivamente, las acciones que se espera conducirán a una situación que mantendrá los parámetros positivos asociados a la situación actual. La Fig. 2.25 representa una *Emoción Positiva* y el proceso que ésta motiva, y que produce un *Comportamiento de Continuación*.

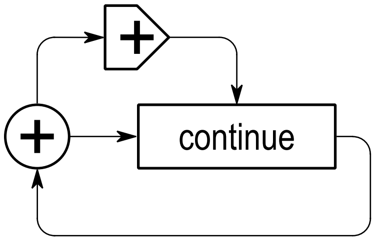

Fig. 2.25 Emociones positivas y comportamientos de continuación o permanencia

Emociones negativas y comportamientos de escape

Cuando la actual situación se aprecia como no favorable, una *Emoción Negativa* motiva un *Comportamiento de Escape* de la situación actual. Este comportamiento consiste en realizar activa o pasivamente las acciones que se espera conducirán a una situación que no presenta los parámetros negativos asociados a la situación actual.

La Fig. 2.26 representa una *Emoción Negativa* y el proceso que ésta motiva, y que produce un *Comportamiento de Escape*.

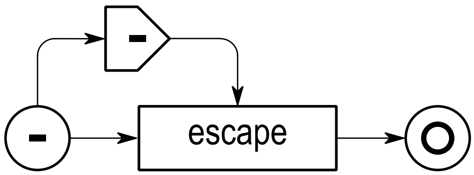

Fig. 2.26 Emociones negativas y comportamientos de escape

La nueva situación esperada podrá pre-apreciarse a su vez como favorable, desfavorable o indiferente. Los comportamientos que se motivarán dependerán de los métodos de escape disponibles, y que sean aplicables dada la situación actual. La motivación del comportamiento de escape será función de la comparación entre las apreciaciones de las situaciones actual y deseada: negativa frente positiva, negativa frente negativa comparada, negativa frente neutra con medida de riesgo.

Emociones positivas y comportamientos de prosecución

Independientemente de la apreciación favorable, desfavorable o indiferente de la situación actual, una nueva situación favorable puede ser visualizada y pre-apreciada. Una *Emoción Positiva* motiva un *Comportamiento de Deseo y Prosecución* de la nueva situación imaginada (situación deseada).

La Fig. 2.27 representa una *Emoción Positiva* y el proceso que ésta motiva, y que produce un *Comportamiento de Deseo y Prosecución*.

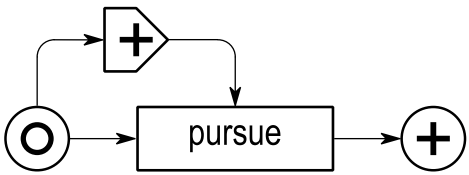

Fig. 2.27 Emociones positivas y comportamientos de prosecución

Los comportamientos que se motivarán dependerán de los métodos que sean aplicables dada la situación actual, y la comparación entre las apreciaciones de las situaciones actual y deseada: negativa frente a menos negativa, positiva frente a más positiva, neutra frente a positiva.

La situación actual podía ser una situación positiva, con lo que la motivación del comportamiento de prosecución debe competir con la motivación del comportamiento de permanencia en el caso de que dicha emoción de permanencia se haya activado (se haya podido evaluar la valoración positiva de la situación actual). La situación actual podría ser también una situación negativa, con lo que la motivación de comportamiento de prosecución competirá con la motivación del comportamiento de escape.

Conflictos y arbitraje

Los comportamientos de escape tienen muchas veces “dudas motivacionales” si no hay garantías de llevar a situaciones más favorables.

El conflicto entre motivaciones se resuelve por comparación y diferencias. Cuando las diferencias no son significativas, las acciones pueden resultar contradictorias. En ese caso, la emoción intrínseca frente a la apreciación de la situación de conflicto entre los accionamientos, puede motivar un comportamiento de arbitraje, aplicando un método reactivo breve, que permita salir del impás.

Emociones neutras proactivas y comportamiento de investigación

Son emociones relacionadas con situaciones desconocidas y que requieren de una investigación adicional antes de poder apreciarlas como situaciones favorables o desfavorables.

Cuando la situación actual no puede ser apreciada como positiva o negativa, una emoción de curiosidad motiva un comportamiento de investigación, que dedica recursos a observar adicionalmente la situación para obtener una apreciación más concluyente.

La Fig. 2.28 representa una *Emoción Neutra Proactiva* y el proceso que ésta motiva, y que produce un *Comportamiento de Investigación*.

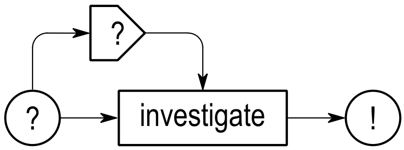

Fig. 2.28 Emociones neutras proactivas y comportamiento de investigación

Este proceso puede: (1) desembocar en una valoración positiva o negativa y desencadenar los comportamientos subsecuentes asociados, o (2) continuar con la indefinición de la apreciación. Una emoción intrínseca de apreciación de las situaciones indeterminadas de forma permanente puede generar un estado emocional de tipo negativo (nivel 1: ansiedad-estrés, nivel 2: tristeza-depresión) asociado a la permanencia prolongada en una indefinición sobre la situación actual.

Ejemplo de una emoción proactiva – sorpresa

Ante una sorpresa, cuando todavía no se ha podido establecer una apreciación de la situación como positiva o negativa, nuestra respuesta suele consistir en abrir más los ojos, para adquirir más información e investigar más rápido y mejor. Posteriormente, el comportamiento de investigación ante la sorpresa puede conducirnos a una re-apreciación positiva o negativa de la situación, con la activación de emociones de miedo, o alegría. Con el miedo, continuamos con los ojos muy abiertos para planificar rápido una respuesta de escape. Con la alegría, solemos pasar a relajar los ojos, quedando confiados en la situación.

Emociones Negativas y Comportamientos de Evitación

Una situación actual, no necesariamente positiva ni negativa, puede ser observada desde un punto de vista de proyección en el futuro, de forma que la previsible situación futura se aprecie negativamente (visualización de una situación no deseable). En ese caso, una emoción de evitación puede motivar un comportamiento que modifique las condiciones para que la situación indeseada no se alcance.

La Fig. 2.29 representa una *Emoción Negativa* y el proceso que ésta motiva, y que produce un *Comportamiento de Evitación*.

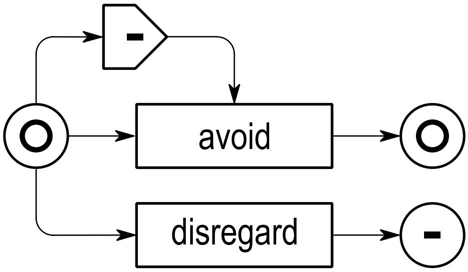

Fig. 2.29 Emociones negativas y comportamientos de evitación

Apreciaciones de la emoción intrínseca

aRTEA propone el siguiente conjunto de apreciaciones emocionales que contribuyen a la emoción intrínseca. Son apreciaciones sobre la situación el propio proceso de resolución del problema, independientemente del problema específico.

- Apreciación de Importancia (beneficio obtenible)
- Apreciación de Confianza (expectativa de éxito)
  - En la percepción del estado actual del problema
  - En el conocimiento sobre la dinámica del problema
  - En los métodos de resolución para el problema
  - En la disponibilidad de recursos de procesamiento
- Apreciación de Urgencia (distorsión por plazo)
- Apreciación de Oportunidad (distorsión por balance esfuerzo-recompensa)

Apreciación de importancia

Valora la importancia de resolver el problema según la escala de valores general del agente. El nivel de satisfacción del problema una vez resuelto contribuye al nivel de felicidad del agente en función de la apreciación de importancia.

A partir de la escala de valores, que determina las importancias de los tipos de problemas genéricos, es determinada para cada problema mediante un proceso de propagación, en paralelo con proceso de descomposición del problema en subproblemas.

Ambos procesos son procesos operativos que se ejecutan de forma periódica.

Apreciación de confianza

Las apreciaciones de confianza, en sus diferentes variantes, son necesarias para hacer intervenir el proceso de resolución del problema, más allá de la simple valoración de qué beneficios se obtendrían de estar en la situación deseada cuando el problema fuera resuelto.

Gran parte de las decisiones emocionales del agente tienen que ver con su visión realista de sus propias capacidades. Las apreciaciones de cada uno de los componentes que intervienen en el proceso de resolución del problema permiten ponderar de forma más equilibrada los distintos aspectos de dicho proceso.

Esta discriminación de los distintos aspectos abre la puerta a tomar medidas en cada uno de los frentes abiertos. Por ejemplo, si la confianza decae en los aspectos relacionados con los modelos mentales del problema, su dinámica o los métodos de resolución, la estrategia podría consistir en reforzar los procesos de aprendizaje. Si por el contrario la confianza decae en los aspectos relacionados con los recursos de procesamiento y su disponibilidad, la estrategia pasaría por iniciar procesos de provisión de recursos materiales.

Apreciación de urgencia y oportunidad

Las apreciaciones de urgencia y oportunidad pueden verse como distorsionadores del esquema de importancias. Un problema de urgente resolución se percibe subjetivamente como más importante. La estrategia de dirigir el comportamiento bajo el influjo de la urgencia puede parecer una estrategia irracional o errónea, sin embargo es una estrategia muy aplicada por los agentes naturales, con lo que hay que permitir al aRTEA poder considerar estas apreciaciones y ver su efecto en el desempeño en la resolución de distintos tipos de problemas. Hay tipos de problemas, que por su naturaleza (dimensión, coste de resolución, plazo, importancia, etc.) en los que la estrategia de considerar la urgencia para provocar una inversión de prioridades es productiva y otros en los que los efectos serán perjudiciales.

La oportunidad es una medida de la recompensa a corto/largo plazo frente a un esfuerzo a corto plazo. La recompensa puede surgir de la simplificación del esquema de problemas, en la que los recursos de procesamiento pueden quedar liberados en el futuro, con lo que problemas no tan importantes, pueden resolverse con objeto de conseguir dicha simplificación. Una segunda causa de beneficio puede estar en los plazos del problema que se atiende como una oportunidad, que podrían llevarlo a ser un problema urgente en el futuro próximo con el aparente aumento de su importancia si el agente es emocional frente a la urgencia. Las satisfacciones de resolución de problemas son también una causa de modificación del estado emocional general del agente. Un ratio elevado de resolución de problemas en el tiempo, suele ser causa de felicidad.

2.3.5. Sistema de atención

El *Sistema de Atención* (Fig. 2.30), define los procesos de selección de comportamientos. Considera el nivel de motivación de los comportamientos generado por el *Sistema de Emoción* y aplica una política de reparto de recursos de procesamiento.

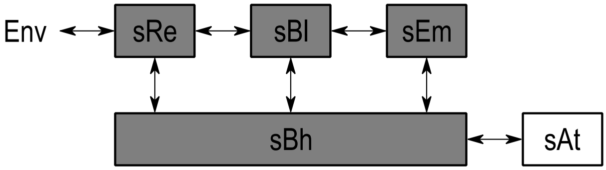

Fig. 2.30 Sistema de atención

El problema de la atención

El proceso de atención consiste en dedicar los recursos disponibles a unos problemas en detrimento de otros. Se trata de un problema de decisión.

Las teorías de la utilidad y del coste de oportunidad, han considerado la forma en la que las elecciones de alternativas afectan a la recompensa obtenida. Estos conceptos, muy desarrollados en el área de la economía, son de aplicación general a los problemas de decisión.

En aRTEA se prestan recursos de procesamiento a un problema con la esperanza de resolverlo. Debido a que los recursos de procesamiento son limitados, y a que en general los problemas que van surgiendo superan la capacidad del agente, éste debe realizar un reparto de sus recursos entre los problemas activos. Esto conduce a un problema de decisión multivariable, donde no es sencillo establecer una lista de criterios, ya que deben ponderarse los efectos de priorizar unas alternativas frente a otras teniendo en cuenta los efectos de cada decisión en cada una de las variables. Esta valoración es muy dependiente de la aplicación.

Características de la atención en aRTEA

A continuación se detallan las características principales del proceso de atención en aRTEA.

Ámbito del proceso de atención - procesador

El modelo de atención en aRTEA está basado en el concepto de procesador. aRTEA define un subproceso de atención para cada elemento significativo de procesamiento. El nivel de granularidad del proceso de atención, establecido por el nivel de significación de cada elemento de procesamiento, puede ajustarse a elección de bRTEA.

Mecanismo de atención

El mecanismo consiste en:

1. Distribuir la carga entre los distintos procesadores que integran el sistema de procesamiento del agente, lo que consiste en asignar subconjuntos de problemas a cada procesador, según una política de distribución de la carga.
2. Dedicar cada procesador a ciertos problemas de los problemas que le han sido asignados, según su motivación, y según una política de asignación de dedicación.

Centralización-descentralización de la decisión

El proceso de atención en aRTEA es, por una parte, un proceso centralizado, ya que debe encargarse del reparto de unos recursos comunes, y por otra, un proceso descentralizado, ya que la aplicación de los criterios de dedicación debe realizarse en base a las características de cada problema, y la evaluación de dichas características puede hacerse con la información local de cada uno de los problemas.

Para poder tomar la decisión, bRTEA debe definir, de forma explícita, un conjunto de modelos para cada uno de los tipos de problemas. Estos modelos constituyen la interfaz de los comportamientos con el *Sistema de Atención*. Estos modelos soportan la componente distribuida (descentralizada) del proceso de atención.

Adicionalmente debe definir un modelo centralizado que codifique la política de reparto de recursos.

Modelos explícitos descentralizados – a nivel del problema

El mecanismo de atención en aRTEA aplica tres modelos que deben definirse de forma explícita en cada pensamiento de resolución de problema:

- Modelo de Satisfacción – Definido como parte del Deseo que genera el Problema. Establece una valoración para la satisfacción de las situaciones. En aRTEA, un deseo no se formula como una situación concreta, sino como un espacio de situaciones posibles. El modelo de satisfacción valora cada situación en dicho espacio de situaciones.
- Modelo de Dinámica – Que codifica la dinámica del problema en la forma de un proceso de cambio de situaciones. Este modelo define las condiciones que permiten el cambio.
- Modelo de Dedicación – Que establece la dedicación de procesador necesaria para alcanzar una situación objetivo dadas unas condiciones de riesgo limitado.

Para permitir el correcto desempeño del proceso de atención, limitando el coste computacional, cada uno de estos modelos debe poderse interrogar mediante procesos reactivos (de tiempo de respuesta acotada), lo que puede exigir que estén basados en métodos simplificados que den respuestas aproximadas.

Modelo centralizado – política de reparto de recursos

El modelo de política de reparto de recursos define el criterio básico de decisión del *Sistema de Atención*.

aRTEA propone utilizar un modelo *Motivación-Satisfacción*. En este modelo, el nivel de motivación del problema establece el nivel de satisfacción que el agente debería intentar alcanzar como solución al problema. El modelo se define mediante una curva, cuya forma indica el tipo de política.

- Políticas de tipo *Todo-para-el-más-motivado*, conducen a enfocar los recursos en los problemas más motivados, y dejar sin atender los problema menos motivados. Los niveles de satisfacción esperables en la resolución de los problemas más motivados son altos, pero nulos en los menos motivados.

- Políticas de tipo *Un-poco-para-cada-uno*, conducen a un reparto más uniforme de los recursos, con niveles de satisfacción esperables en la resolución del conjunto de los problemas más bajo.

El resultado de aplicar una política es un nivel de *Felicidad* para el agente, obtenido como la media de los niveles de satisfacción/frustración del conjunto de problemas.

Método de decisión – negociado de dedicación

El método de decisión es un proceso iterativo en el que se asignan recursos a los problemas hasta agotar la capacidad del procesador.

El reparto se realiza por el nivel de motivación de los problemas, y la dedicación asignada a cada uno de ellos se determina mediante un proceso interrogativo de tres pasos que el *Sistema de Atención* realiza sobre cada problema. Este proceso se muestra gráficamente en la Fig. 2.31.

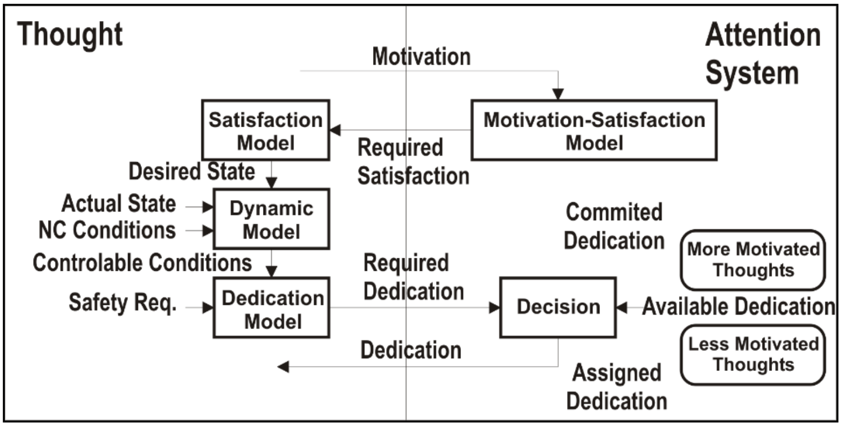

Fig. 2.31 Negociado de dedicación

Los pasos en el negociado de dedicación son los siguientes:

1. Mientras el procesador disponga de capacidad, se considera el siguiente problema más motivado.
2. El nivel de motivación de dicho problema establece el *Nivel de Satisfacción Esperado* para su resolución, aplicando el *Modelo de Motivación-Satisfacción*.
3. El *Nivel de Satisfacción Esperado* establece una *Situación Deseada Específica* del espacio de situaciones definido por el deseo, aplicando el *Modelo de Satisfacción* y un criterio adicional de selección que depende del problema.
4. La *Situación Deseada Específica* establece las condiciones que el agente debe establecer para producir el cambio de situación, aplicando el *Modelo de Dinámica* del problema.
5. Como parte de las condiciones controlables para dicho cambio, se determina la *Dedicación Requerida* de procesador para dicho problema, aplicando el *Modelo de Dedicación* y el criterio de seguridad y de limitación del riesgo que establece el problema.
6. La *Dedicación Requerida* es solicitada al procesador.
7. Finalmente la dedicación es asignada, o bien inmediatamente, o bien es puesta en una lista de peticiones pendientes, en el caso de que haya cambios en el esquema de reparto de dedicación y alguno de los problemas actualmente activos deba perder dedicación de procesador (y según el plazo de desatención que su modelo dinámico establezca para el criterio de seguridad)

2.3.6. Sistema de relación

El *Sistema de Relación* (Fig. 2.32), define los dispositivos y los procesos que permiten a RTEA interactuar con su entorno.

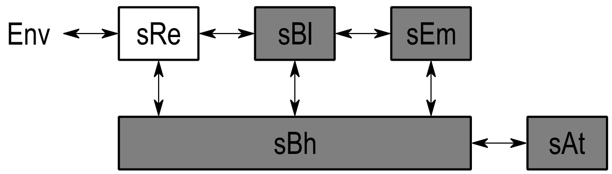

Fig. 2.32 Sistema de relación

Los procesos de relación intervienen de forma esencial en el comportamiento exteriorizado o efectivo.

Los procesadores de relación son dispositivos especializados que incorporan elementos transductores para enlazar ciertas magnitudes del entorno con los conceptos en el sistema de creencias.

Sentido del flujo de información

El convenio de sobre el sentido del flujo de información, centrado en el punto de vista del agente, establece que las *Entradas* constituyen la información que fluye desde el entorno hacia la mente del agente, mientras que las *Salidas* constituyen el flujo de información desde la mente del agente hacia el entorno.

Sentidos

Los *Sentidos* son los dispositivos que permiten percibir el entorno. Obtienen información en crudo.

Las *Percepciones* son conceptos que representan las capturas de información de los sentidos.

Conjunto de sentidos

Cada implementación de RTEA define un conjunto de sentidos con unas capacidades diferentes. La selección de sentidos es definida por la aplicación.

Propiedades de los sentidos

Definen las magnitudes que pueden percibir, sus características de sensibilidad, precisión, alcance, foco, disponibilidad, fiabilidad, eficiencia, coste energético, tiempo de respuesta y fatiga.

Efectores

Los *Efectores* son los dispositivos que permiten manipular el entorno. Utilizan acciones motoras elaboradas desde procesos de reacción/deliberación.

Las *Acciones Motoras* son conceptos que representan la acción del sistema de relación sobre el entorno.

Conjunto de efectores

Cada implementación de RTEA define un conjunto de efectores con unas capacidades diferentes. La selección de efectores es definida por la aplicación.

Propiedades de los efectores

Definen las magnitudes que pueden manipular, sus características de potencia, sensibilidad, precisión, alcance, foco, disponibilidad, fiabilidad, eficiencia, coste energético, tiempo de respuesta y fatiga.

Canales de comunicación

Son dispositivos especializados que permiten una entrada/salida de información representada con un lenguaje. Es información ya elaborada, pero que el agente puede producir y consumir para relacionar su mente con el entorno.

Conjunto de canales de comunicación

Cada implementación de RTEA define un conjunto de canales de comunicación con unas capacidades diferentes. La selección de canales de comunicación es definida por la aplicación.

Propiedades de los canales de comunicación

Definen el tipo de información que puede ser compartida, el nivel de interpretación, el lenguaje utilizado, el canal físico, la fiabilidad de la información intercambiada y la disponibilidad

Procesos de relación

Los siguientes son los tipos de procesos implementados por el Sistema de Relación:

- *Procesos de Percepción* – Son procesos que generan Percepciones a partir de la actividad de los Sentidos. Estas percepciones informan sobre la situación del entorno. La percepción es información en bruto. En ese sentido, puede ser considerada como información, más que como conocimiento elaborado para algún contexto de problema específico. Hay sin duda una interpretación de las magnitudes del mundo real que los sentidos perciben, ya que en el proceso de percepción se aplica un principio de transducción basado en algún modelo o función de transferencia. Sin embargo, el grado de interpretación es tal que no se hace ningún supuesto sobre el uso que de la información generada se va a realizar, por lo que puede considerarse a la percepción como información genérica (multiuso) sobre el entorno.
- *Procesos de Fusión de Percepción* – Son procesos que integran en una única percepción información proveniente de varios canales de percepción (sentidos y/o canales de comunicación de entrada, bien desde instrumentos de percepción, bien desde otros agentes compartidores de información).
- *Procesos de Observación* – Son procesos de interpretación de las percepciones en el contexto de un problema dado. Pueden estar basados en percepciones u otras observaciones previas.

- *Procesos de Fusión de Observación* – Las observaciones también son, en general, procesos de fusión, de forma que se interpretan en conjunto varias fuentes de información.
- *Procesos de Acción Motora* – Son procesos que permiten manipular el entorno real a través de los efectores (motores)
- *Procesos de Acción* – Las manipulaciones que el agente realiza en el entorno pueden ser el resultado de dos tipos de procesos: reacciones y deliberaciones. Las acciones son procesos de acción resultado decisión en el transcurso de una deliberación, en el contexto de un pensamiento para la resolución de un problema.
- *Procesos de Reacción* – Otras acciones motoras son el resultado de aplicar un esquema de respuesta reactivo, de baja latencia, para responder a situaciones que requieran un tiempo de respuesta reducido.
- *Procesos de Fusión Motora* – En general, los efectores pueden ser solicitados por más de una acción o reacción simultáneamente, con lo que los procesos de Acción Motora suelen ser procesos de fusión de información desde varias fuentes. En el proceso de fusión se deben aplicar políticas para la integración de las acciones motoras (acciones desde deliberaciones, o reacciones) de forma que las manipulaciones que el agente realice en su entorno sean consistentes.

3. Publicación A

Multicore and FPGA Implementations of Emotional Based Agent Architectures

Carlos Domínguez, Houcine Hassan, Alfons Crespo, José Albaladejo.

Journal of Supercomputing. 71(2) 479-507

DOI: 10.1007/s11227-014-1307-6

Abstract

Control architectures based on Emotions are becoming promising solutions for the implementation of future robotic agents. The basic controllers of the architecture are the emotional processes that decide which behaviors of the robot must activate to fulfill the objectives. The number of emotional processes increases (hundreds of millions/s) with the complexity level of the application, limiting the processing capacity of the main processor to solve complex problems (millions of decisions in a given instant). However, the potential parallelism of the emotional processes permits their execution in parallel on FPGAs or Multicores, thus enabling slack computing in the main processor to tackle more complex dynamic problems. In this paper, an emotional architecture for mobile robotic agents is presented. The workload of the emotional processes is evaluated. Then, the main processor is extended with FPGA co-processors, which are in charged of the parallel execution of the emotional processes. Different Stratix FPGAs are compared to analyze their suitability to cope with the proposed mobile robotic agent applications. The applications are set-up taking into account different environmental conditions, robot dynamics and emotional states. Moreover, the applications are run also on Multicore processors to compare their performance in relation to the FPGAs.

Experimental results show that, Stratix IV FPGA increases the performance in about one order of magnitude over the main processor and solves all the considered problems. Quad-Core increases the performance in 3.64 times, allowing to tackle about 89% of the considered problems. Stratix III could be applied to solve problems with around the double of the requirements that the main processor could support and a Dual-Core provides slightly better performance than Stratix III and it is relatively cheaper.

3.1. Introduction

Many research works [1, 2, 3, 4] predict a growth of the number of intelligent robots in the industry and in our lives in the two next decades. They state that advanced robots capable of making decisions on their own as humans do are still under development and the first prototypes will not start to appear until 2030. Some researches [2, 6] state that we are seeing the emergence of the first generation of robots such as the demining robot Warrior manufactured by iRobot [7], which are only able to solve simple tasks with little ability to adapt to the changing environment, and running their program code on a single-core processor. However, more intelligent features that robots would include, such as decision-making, are not yet developed in real robotic agents. It is expected that by 2050, these agents will be implemented in advanced computers capable of running hundreds of billions of instructions (i.e., 4th. generation). These robots would rival human intelligence and would be able to perform operations of abstraction and generalization, medical diagnostics, planning and decision-making [5, 24, 25].

These kinds of applications involve high complexity, such as the proposed multi-purpose mobile robotic agent performing transportation, diagnosis, cleaning, and surveillance services simultaneously in uncertain and unpredictable environments. Each service problem is decomposed in a set of sub-problems and possible alternatives to be assessed (e.g., observation, path planning and object handling sub-problems for the transportation service). In the same way, each of the sub-problems is decomposed in simpler tasks (e.g., the path planning sub-problem generates a full tree of path alternatives that must be assessed to select one path). As this decomposition is performed for each of the sub-problems of all the simultaneous services, it arises that in a given instant the total number of decision alternatives that the agent has to manage are significantly high (e.g., 1M decisions).

On the other hand, control architectures based on emotions are inspired on emotional natural agents. They are becoming promising solutions for the implementation of advanced robotic systems [8, 9, 10] because they facilitate the process of decision-making [17, 18]. They use the mechanism of emotions to organize the behaviors. It has the following advantages: (i) allow the robot to be autonomous to focus its attention on the most promising behavior, (ii) provide a bounded response time which helps organizing the deliberative processes, (iii) sort the problems based on the expectations of success, (iv) autonomously adapt the computational load to the available processor capacity allowing solving problems of increasing complexity, (v) separate the decision from the action processes and the use of subjective appraisal of the situation permit find always an alternative solution.

In this paper, an emotional robotic architecture for the control of complex mobile robot applications is used. In this programming model, there coexist two main types of processes: behavior and emotion processes. The formers solve the application problems (e.g., surveillance) while the latter use an emotional mechanism to motivate the robot behaviors.

Originally, all the processes of the emotional architecture, including the behaviors and the emotions were executed on a single-core computer (e.g., AMD 3.3GHz). The emotion processes must be applied to all problems/subproblems of the agent agenda at every cycle of attention (e.g., 0.1s) to decide which behaviors to execute. As the number of problems grows in the agenda, the emotional workload increases significantly as well. For instance, the proposed multipurpose robotic agent, to tackle complex problems, will need a high computational workload of about 200 Millions emotion Operations per Second (MOPS), as will be shown in the experiments section. Where an Operation is a Dot-Product function involving: a hyperbolic tangent function, a multiplication by a weight and a sum of up to 6 other functions computed in parallel. However, the single-core computer cannot support this workload because it can only execute 25 MOPS. Moreover, the implementation of the emotion processes on an MCU or low-medium performance DSP is discarded also because these devices provides even less power computation (i.e., between 10 MOPS and 20 MOPS). Therefore, the most suitable solutions that provide the performance required by the above-mentioned applications are high performance FPGA and Multicore processors. To this end, we transfer the execution of the emotion processes to the FPGA, thus the single-core computer gets slack time to solve more complex applications by: (i) improving the productivity of the simultaneously number of problems, or (ii) tackling more time critical dynamic problems (e.g., solve the problems at highest speed).

Regarding the FPGA processors, the communication between the

FPGA and the Control Computer (CC) is performed through Ethernet. The CC sends, at each attention cycle, the parameters of the emotions through an Ethernet connection. The FPGA performs the calculation and sends back the results of the behaviors' motivation to the CC. As will be presented in section 3.3.2, the emotional process has an accumulative phase. Thus, in order to optimize the use of the processing resources of the FPGA-based coprocessor, the CC arranges the parameters of the emotions over the data flow in a specific order. The latencies of each of the phases in the processing pipeline of the FPGA require this optimal order. Different FPGA models have been analyzed. Low performance FPGA are not sufficient to tackle the execution of the emotion processes due to their limited computational power (e.g., Stratix EP1S20F484I6 performs only 15 MOPS). Therefore, high performance FPGA Stratix III and Stratix IV [12, 13] are selected. To undertake the implementation, the potential parallelism of the emotion processes is identified and characterized. The Dot-Product functions based on hyperbolic tangent that are computed in parallel are optimized based on the A3 methodology [19] to use the optimal number of emotion operators. Finally, the emotional processor is designed in VHDL for both FPGA models.

A second implementation of the emotional architecture is carried out using Multicore processors in order to compare the performance of the FPGA's and the Multicore, depending on the number of dedicated cores. In this case, a six-Core Intel i7 processor is used [14]. Four of the cores are dedicated to the emotion processes, one for the application processes and one for the attention system. The emotion processes are allocated to the different cores using the Worst Fit algorithm, which allows balancing the total workload among the cores. Each core implements a local Rate Monotonic Scheduler to support the execution of the processes [20].

In the experimental evaluations, different application problems of varying complexity levels (simple, normal, complex), under distinct environmental conditions and robot agent dynamics (safe, normal, risky), and different emotional states (relaxed, normal, stressed), are implemented using the FPGAs. The obtained performance results are compared with the execution of the same applications on the multicore processor. Experimental results show that, Stratix IV FPGA increases the performance in about one order of magnitude over the main processor and solves all the considered problems. Quad-Core increases the performance in 3.64 times, allowing to tackle about 89% of the considered problems. Stratix III could be applied to solve problems with around the double of the requirements that the main processor could support and a Dual-Core provides slightly better performance than Stratix III and it is relatively cheaper.

The rest of the paper is organized as follows: section 3.2 reviews the state of the art of emotional architectures and their implementations; section 3.3 describes the general characteristics of the emotional control architecture and details the emotional system; section 3.4 describes the multipurpose robotic agent application and the FPGA and Multicore based emotional processor designs; experimental evaluation and results are discussed in section 3.5; finally, conclusions are sum-up in section 3.6.

3.2. Related work

Different control architectures based on emotions are found in the bibliography. Arkin et al. [17] develop algorithms based on the emotion of deception to control robotic agents. The authors are inspired on the behavior of deception observed in animals or humans. In the simulations they show that robots including this emotion are more effective. Salichs [18] proposes a decision-making system based on drives and motivations, but also emotions and self-learning. The agent's goal is to learn to behave through interaction with the environment, using reinforcement learning, and maximizing their welfare. Lee-Johnson et al. [8] develop a hybrid architecture reactive/deliberative that incorporates artificial emotions to improve decision-making and actions of mobile robotic agents. Emotions are active at different levels of the architecture and serve to modulate the decisions and actions of the agent. Damiano [2] presents a model where decision-making is based on a motivational system. The motivations are dependent on the value of the need that have to be satisfied and a stimulus incentive. Once they calculate all the values, the highest motivation activates and organizes the behavior so as to satisfy the most urgent need. On the other hand, intelligent agents have been implemented using different SoC technologies. In [11] a neuro–fuzzy agent for ambient-intelligence environments is proposed. The agent has been implemented as a system-on-chip (SoC) on a FPGA around a MicroBlaze processor and a set of parallel intellectual property cores for neuro–fuzzy modeling. The SoC is an autonomous electronic device able to perform real-time control of the environment in a personalized and adaptive way, anticipating the desires and needs of its inhabitants. In [15] authors present a parallel genetic programming (PGP) Boolean synthesis implementation based on a cluster of Virtex5 FPGAs using parallel programming and hardware/software co-design techniques. The performance of the cluster of FPGAs implementation has been compared with an HPC implementation resulting in an improvement of the speed up and in terms of solving the scalability problems of this algorithm. A practical implementation of a neural network based estimator of the load machine speed for a drive system with elastic coupling, using an FPGA placed inside the NI CompactRIO controller is presented in [16]. The algorithm code for the neural estimator implemented in C-RIO was performed using the LabVIEW software. The focus is on the minimization of the used programmable blocks of the FPGA matrix. Tests of the load machine estimator implementation are performed and results show high-quality state variable estimation of the two-mass drive system.

The aforementioned papers present very interesting implementations of complex control applications, however, regarding the implementation of emotional models, this is for the best of our knowledge, the first proposal of a parallel implementation of emotional architectures on FPGA and Multicores. Moreover, this paper differs from the above ones in the sense that it tackles the NP-hard problem of decision making in multi-objective intelligent agent applications, where different high-level complexity problems are simultaneously solved using an emotional system. In the proposed system, the computational power of the applications is higher compared to the previously commented papers, due to the high number of problems and decision alternatives to undertake. These types of problems are very different from the above ones, which are more focused in a specific well-defined task (i.e., robot manipulator) where the number of solution alternatives is small. Likewise, the paper uses high performance FPGAs: Stratix III and IV to tackle the implementation of the emotional robotic architecture and provides a discussion of the experimental results regarding the convenience of implementing the emotional model whether on an FPGA or on a Multicore processors.

Some ASICs have been developed to give processing support in different areas of artificial intelligence. A remarkable example is the SyNAPSE project [26, 27] lead by IBM, where thousands of neural cores integrated in a single chip offer a processing layer for the emulation of natural neural processes. The model of the emotion process in our agent architecture, however, differs from the artificial neural network model; therefore, the emotions have been implemented on a specific processor. The development of an ASIC for emotional purposes would require considerable financing resources. The initial design phases of a new processing approach typically benefits of the availability and flexibility of the FPGA technology, which is an affordable platform for prototypes and small production series.

3.3. Emotional control architecture

In this section, the complexity of the emotional control architecture is described and the computational requirements of the emotional system and the exhibited potential parallelization are discussed.

3.3.1. Real-time emotional architecture overview

An emotional control architecture has been developed in the group of Industrial Informatics at Universitat Politècnica de València. This architecture (see Fig. 3.1) is composed of five modules: Belief, Behavior, Emotion, Attention and Relation sub-systems. The Belief and Behavior systems represent the application processes, while Emotion and Attention systems are in charged of the execution of the operational processes.

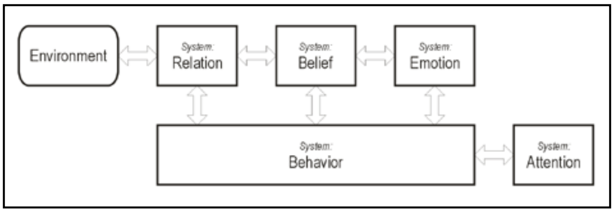

Fig. 3.1 Emotional control architecture block diagram

A behavior of the emotional control architecture is based on a problem resolution process. A problem appears when the agent desires a new situation. Every new desire starts an associated behavior, which defines a context for the execution of observation, decision and action processes related to the problem to be solved. These problem-domain processes are the application processes. On the other hand, the system implements operational processes that use an emotional mechanism to motivate the application processes. Fig. 3.2 shows the information flow in the emotional control architecture. Ellipses represent concepts and squares represent processes. The bold arrows represent the main paths. The (a) path connects sensors and motors devices, from the perception to the action. The application processes in (a) flow through two subways, the deliberative-way (b) and the reactive-way (c). Reactive processes have a guaranteed response time, which is usually short. The response time of deliberative processes however is usually longer. The rest of the paths are used by the operational emotional processes, which generate new desire-behavior (d) and execute motivated behaviors (e).

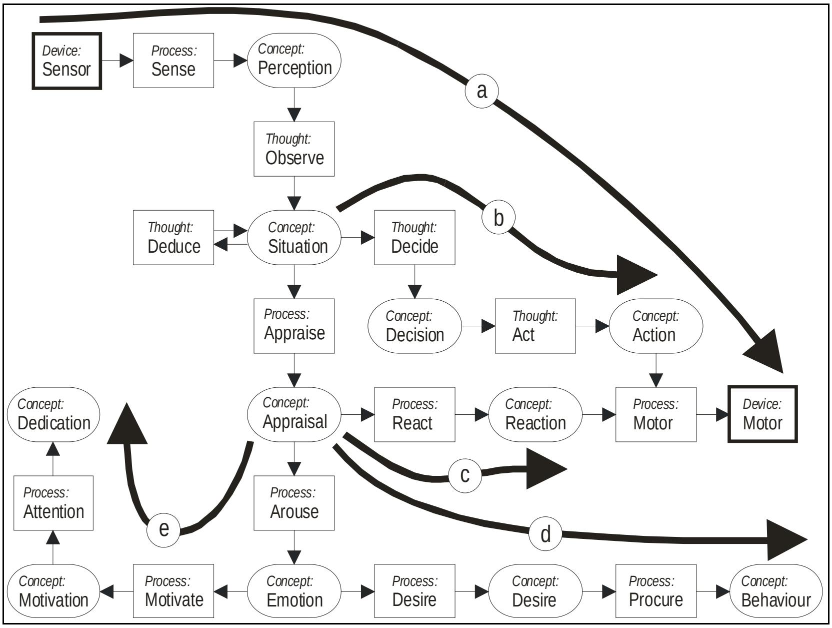

Fig. 3.2 Information flow in the emotional control architecture

Initially, the architecture was running on a main computer (single-core) as shown in Fig. 3.3, where the Input/Output is managed by a DAQ. However, as the complexity of the applications increases the emotional workload raises significantly, reducing the capacity of the single-core to solve the problems. This paper proposes the implementation of the emotional and attention processes on FPGA/Multicore processors to allow the single-core focus in solving more complex problems with high dynamic requirements.

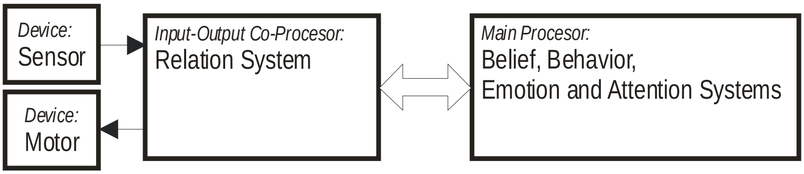

Fig. 3.3 Original system

To show how to perform the distribution of the different types of processes among the single-core and the FPGA/Multicore processors, the structure of the emotional architecture is described.

3.3.2. Emotional processes specification

An emotion is the process of appraising an observed situation and motivating a robotic agent behavior to undertake this situation (see Fig. 3.4).

Fig. 3.4 Emotional process

During its attention cycle, the agent evaluates the set of active emotions. This set of emotions grows and decreases dynamically as new problems are registered/unregistered in the agent's agenda. Two subsets comprise the emotions set: (1) the set of intrinsic emotions and (2) the set of non-intrinsic emotions.

Intrinsic emotions - application independent

One intrinsic emotion is associated to each of the problem instances in the agenda, and its emotional response consists in the motivation of the process in charged of resolving the problem.

The structure of the intrinsic emotions (the number, nature and weight of their emotional contributions) is the same for each instance; that is, it is independent of the problem. The agent builder however, defines this general structure accordingly with the situational factors that are relevant to motivate the problem resolution's process, no matter the specific nature of the problem. These intrinsic emotion structure definitions define and name different agent characters.

Example of intrinsic emotions

In the case of the mobile robot's application presented in paragraph 3.4.1, the Importance, Confidence, Urgency and Opportunity situation appraisals, contribute to the intrinsic emotions. The Importance emotional contribution comes from the appraisal of the benefits that would be obtained if the problem is resolved, meanwhile the remainder emotional contributions come from the appraisal of the success' expectative on the problem being solved. The Confidence emotional contribution has different dimensions: Confidence on the situational observations, Confidence on the problem solving method, and Confidence on the processor availability. The Urgency and Opportunity contributions distortion the importance of the problem, and contribute to the motivation of each problem, at least partially, in an inter-problem basis, causing motivation inversions; the Urgency dealing with the deadlines and the adverse effects of not meeting them; and the Opportunity dealing with the benefit-effort balance at each time.

Non-intrinsic emotions - application dependent

The methods that the problem-solving processes apply follow a sub-problem decomposition strategy with a sequence-and-alternative schema.

Every new sub-problem built and registered in the agenda is the result of an emotional decision. When a method is defined, either during the agent initial building or during a later learning process, non-intrinsic emotions are defined and linked to key emotional decision points in the method. These non-intrinsic emotions are application dependent, so their structure (the situation appraisals that contribute to the emotion activation, the contribution functions, and their weights) are specific for the type (not at instance level) of problem. The agent builds and registers the non-intrinsic emotions when the method reaches their specific emotional decision points. After that, the emotions are evaluated every attention cycle, and their response consists of building (or destructing) new sub-problem resolution processes, giving them an importance level (applying an importance appraisal propagation mechanism), and building and registering their intrinsic emotions.

A key design decision of the agent builder is the level of granularity of the decomposition of the problem. The agent builder explicitly expose this decomposition to the emotional motivation mechanism. Beyond it, the methods applied in the final steps of the problem decomposition look like black boxes to the emotional system.

Example of non-intrinsic emotions

The mobile robot application presented in paragraph 3.4.1 defines a wide set of problem types, resolution methods and emotional decisions. As an illustrative example, consider the Robot Displacement Problem. Since most of the services require the robot to get to some spatial locations, the displacement problem arises often, and several, or even many, instances of the Robot Displacement Problem usually populate the agent agenda at a time. The Displacement Problem decomposes in several sub-problems: path-planning, physical trajectory-planning and motor action. For a path-planning problem, the agent could apply an algorithm to obtain the optimum path on a spaces-accesses' graph representing the robot navigation environment. However, since the navigation space is not fully observable from any current robot location, the displacement plans would usually need to be recalculated while the robot applies them and gets new environmental observations. The emotional approach instead, decomposes the spaces-accesses graph in levels of confidence, e.g. by considering spaces-accesses defined by objects with different location volatility: static objects (e.g. architectural elements), moveable objects (e.g. furniture and machines), and mobile objects (e.g. people and other robots). Then it applies the path finding for each of the subspaces and keeps open different path alternatives while the robot is already moving. The agent searches the spaces-accesses' graph hierarchically. The path-planning problems generate new path-planning sub-problems while their respective sub-plans can be refined. A non-intrinsic emotion (Generate New Path) in the problem resolution method is in charged of this process. When the path's refinement has gotten at an end, a different non-intrinsic emotion (Generate Trajectory) creates a new trajectory-planning problem. Furthermore, the trajectory-planning problems create motor-action-planning problems. The full set of problems in the agenda is arranged on a tree-structure.

Although the agent currently attends only the most motivated problems (until the processing resources get to saturation) the agent must periodically evaluate the full set of emotions to motivate all the problems in the agenda. Thus, the emotional processes cause an extra workload in this architecture. The size of the agenda however, is partially auto-controlled, because in order to create new sub-problems, the parent problems need to get the attention of the processor to reach the emotional decision points, where the new sub-problems are created. Additionally, a proper definition of the agent character permits the control of the emotional sensitivity and limits the emotional state parameter $\varepsilon = t_e / T_a$ (see paragraph 3.3.3)

Despite the extra workload, this emotional approach presents some important benefits. The benefits consist of the explicit separation of the attention process, and the use of an emotional motivation mechanism, which explicitly shows its decision criteria and is configured in a centralized way (defining the agent character). To minimize the impact of the emotional workload the architecture defines a simple reactive model for the emotional motivation processes. This model permits its sequential or parallel execution.

Fig. 3.5 and Fig. 3.6 detail the emotional motivation process. Fig. 3.5 shows the appraisal of the observed situation.

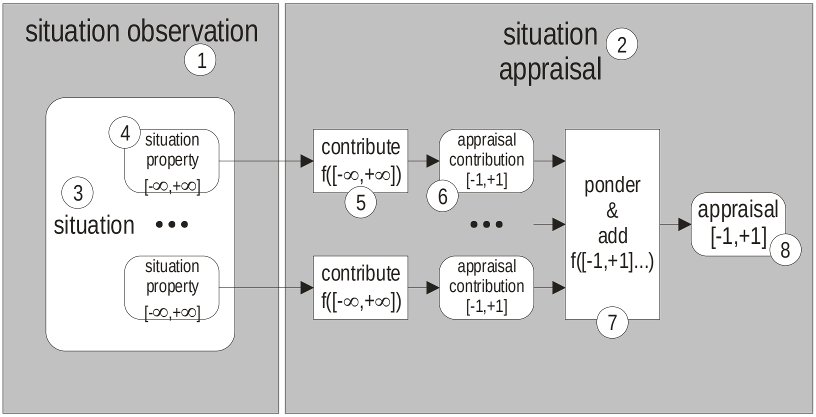

Fig. 3.5 Situation appraisal process

Situations (3) are generated by observation processes (1) and are represented as real properties (4). The appraisal process (2) depends on the agent character. The character dynamically adjusts the parameters of this process. To calculate the appraisal of the situation (8) the agent ponders and adds (7) a set of appraisal contributions (6), which are evaluated using contribution functions (5). The number of situation appraisals for the type of applications considered in the experiments on average is 2M.

Equation 3.1 represents the i^th situation appraisal.

$$a_i = \sum_{k=1}^{l} w_{ak} \cdot f_{ak}\left(p_k\right) \qquad (3.1)$$

Where: p_k is the k^th property of the situation, f_{ak} is the k^th contribution function, w_{ak} is the weight of the function and l is the number of appraisal contribution in the range [1, 6].

Fig. 3.6 shows the emotional motivation process. This process has two phases: first, an emotional activation (9) sets an emotional state (10), and second, an emotional response builds and motivates a behavior (11). The emotional contributions (13), evaluated with contribution functions (12), are pondered and added (14) to finally give an emotional state (15). The emotional contributions functions are defined in the real range [-1,+1] and the emotional state in [0, +1]. Every emotional state is labeled in the robotic agent navigation problem (e.g., “fear of collision”), a 0 level would mean “no fear”, while a 1 level would be “afraid”. The emotional response generates new desires (16, 17), and motivates the behavior to accomplish the desire (18, 19). The number of emotions involved in the proposed robotic applications on average is 0.5M.

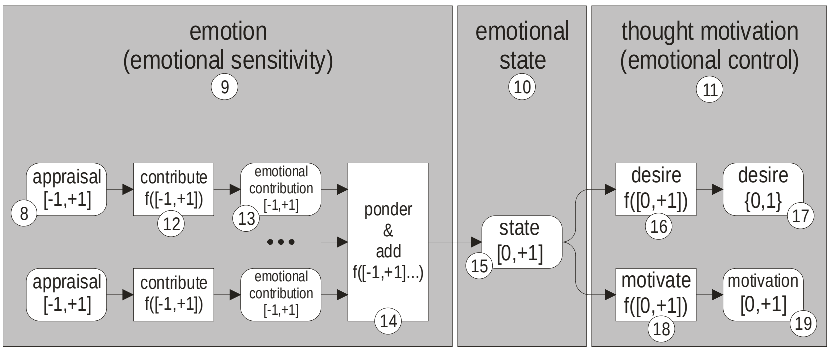

Fig. 3.6 Emotional motivation process

The emotion is expressed in Equation 3.2

$$s_j = \sum_{i=1}^{I} w_{ci} \cdot f_{ci}(a_i) \tag{3.2}$$

Where: s_j is the state of the j[th] emotion, f_{ci} is the i[th] contribution function, w_{ci} is the weight of the function.

The emotion contribution functions, f , have properties such as slight variations at the ends of the range that tend to asymptotic values and abrupt variations around an inflection point in the center of the range. These properties are found in the hyperbolic tangent functions, which are used to represent the contribution functions as shown in Equation 3.3.

$$th(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{3.3}$$

Where x is the appraisal value a_i when calculating the emotion. To allow adjusting the hyperbolic function, Equation 3.3 is transformed in the function shown in Equation 3.4, where the parameters x_0, y_0, δ_x and δ_y permit to translate and scale the contribution function.

$$th^*(x) = \left(\frac{e^{2(x-x_0)\delta x} - 1}{e^{2(x-x_0)\delta x} + 1} - y_0\right)\delta y \tag{3.4}$$

These emotions are grouped in the emotional system shown in Fig. 3.7. The emotional system gets, in a given instant, inputs from a set of N situation appraisals (e.g., 2M) and produces a set of K motivations (e.g., 0.5M). Each emotion can be composed of up to 6 different contributions functions. These contributions have the structure of the Dot-Product functions, each of them consists of: a hyperbolic tangent, a multiplication by the weight, a sum of the different contributions and the identity function. The total number of these Dot-Product functions in the emotional system depends on the complexity of the problem, the environment conditions and the robotic agent dynamics. In the experiments section, applying the multi-objective robotic agent, this number reaches a value of about 200 Million contribution Operations per Second (MOPS).

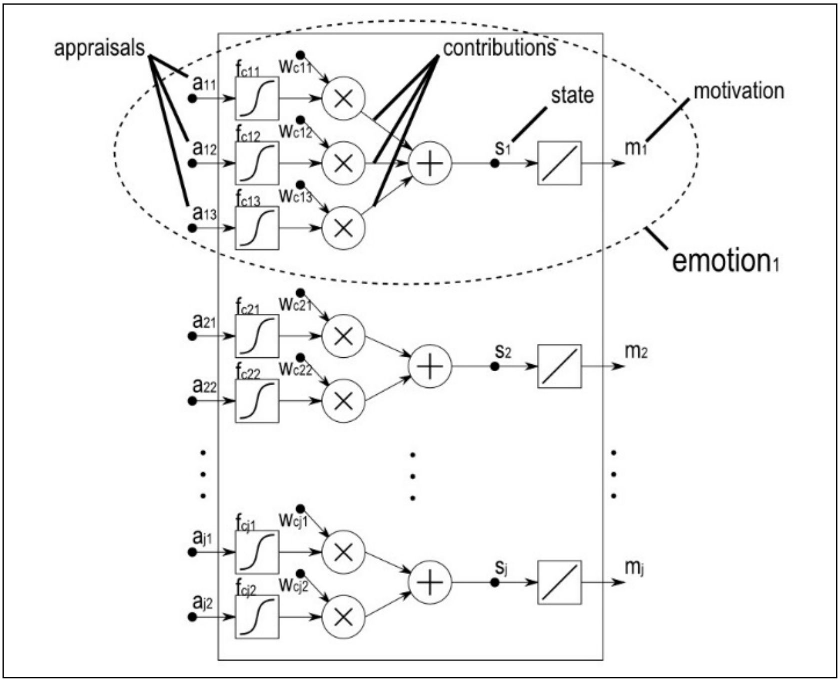

Fig. 3.7 Emotional system structure

These Dot-Product functions can be computed in parallel since they are independent in the sense that they perceive the situation and have to evaluate it and generate an emotional state. However, in the initial implementation they were executed sequentially in a main computer but due to their highly computational requirements, they surpass the capacity of the main processor, leading to the impossibility of executing the rest of the process applications and hence unfulfilling the robotic agent objectives. This situation is analyzed in the next subsection to propose a parallel implementation of the emotional system.

3.3.3. Emotional system computational requirements

The robotic agent executes observation, decision, and action application processes periodically. A period is called the attention cycle and is represented by T_a (see Fig. 3.8). T_a depends on the problem dynamics (e.g., robot speed). Besides of the application processes, the system must execute the operational processes of the emotional system of Fig. 3.7. The temporal cost, t_e , of executing the emotion processes is represented in grey color in Fig. 3.8 while the application processes are represented in white color. The agent needs to balance between the costs of the execution of the application processes and the emotional processes.

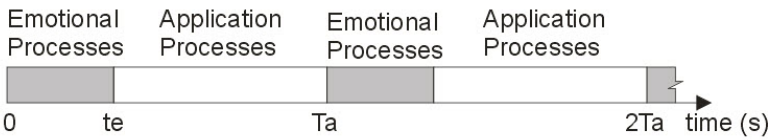

Fig. 3.8 Attention cycle

The workload of the emotional processes strongly depends on the application problem: the definition of the type of services, the complexity of the navigation and operation environment, and the amount of allowed simultaneous service requests.

Firstly, the emotional workload depends on the number of problems in the agenda, which is variable over time. Since the type of the problem defines the number of emotional decision points in the method that resolves the problem, the nature of those problems also affects the workload. The model of the emotional motivation process lets a variable number of emotional contributions, so, the Number of Emotional Contributions, identified as O (for Operations), or MO (for Millions of Operations), is a better choice for estimating the workload than just the Number of Emotions.

Equation 3.5 estimates the workload of the emotional processes in Number of Operations.

$$O = O\left(n_p\left(n_{ic} + n_e \cdot n_{nic}\right)\right) \tag{3.5}$$

Being:

n_p is the number of problems in the agenda.

n_e is the mean of the number of non-intrinsic emotions per problem.

n_{ic} is the number of emotional contributions of the intrinsic emotions, which is a constant value specified by the agent character.

n_{nic} is the mean of the number of emotional contributions of non-intrinsic emotions, which is variable with the type of problems and the distribution over type)

In order to estimate the necessary processor throughput however, it is necessary to calculate the Number of Operations per second. Equation 3.6 estimates the workload of the emotional processes in OPS, being Ta the Attention Cycle Period in seconds.

$$OPS = \frac{O}{T_a} \tag{3.6}$$

Arbitrarily, we have defined three complexity problem levels: Simple, Normal and Complex, related to the number of problems to be processed, which stablish different service performance of the robotic platform, and three risk of robot collision levels: Safe, Normal and Risky, related to the robot navigation speed.

The computational costs of the emotional contributions vary depending on the complexity of the application problems. In the worst case, the number of millions of contributions functions per second is about 200. This growth of the emotional workload can put in danger the accomplishment of the objectives of the application. That is, if the time t_e dedicated to the emotional computation is too high, then the remaining time (T_a - t_e) will not be enough to execute the applications processes. Therefore, the goal is to minimize t_e as much as possible, hence, the robotic agent can dedicate more time to solve practical problems and less time to the emotional processing. To this end, this paper proposes the design of the emotional system in hardware processors.

The selection of the hardware architecture to implement the emotional system will depend on the nature and the volume of the emotional processes. As shown before, the emotions have the Dot-Product functions structure computed iteratively in each attention cycle. This structure is very suitable for its implementation on FPGA processors. On the other hand, the volume of these operations is significantly high (200 MOPS) and cannot be tackled by low-medium performance FPGA. For instance, Stratix EP1S20F484I6 provides only 15 MOPS, which is not enough to cope with the application requirements. These low cost FPGA are used to implement reactive controllers (e. g., manipulators) where the required computational power is low, but they are not adequate in deliberative decision-making processes where millions of alternatives have to be computed in a given instant. Therefore, in this work, Stratix III and IV are used for the implementation of the emotional system. The available resources of Stratix FPGAs permit the synthesis of the communication IP cores necessary to communicate the emotional coprocessor with the main processor, which has been an additional reason for selecting this FPGA family.

A second implementation alternative is the use of Multicore processors. This architecture is also adequate because it permits the distribution of the emotional processes among the cores and their execution in parallel. The high throughput of the multicore will allow reducing substantially the t_e time. A six-core i-7 processor is used to implement the whole architecture. Both implementations are compared to show their performance when undertaking the different robotic agent application problems.

3.4. Emotional processor architecture design

3.4.1. Robotic agent application

The FPGA/Multicore emotional processor is designed to tackle robotic agent applications as shown in Fig. 3.9. The multipurpose robotic agent performs activities such as diagnosis, transportation, cleaning, and surveillance simultaneously. Initially, the single-core (control computer) was executing the whole workload of the robotic agent (i.e., application processes and emotional processes). In the proposed design, the emotional processes are transferred to the FPGA/Multicore to allow the single core solve more complex problems.

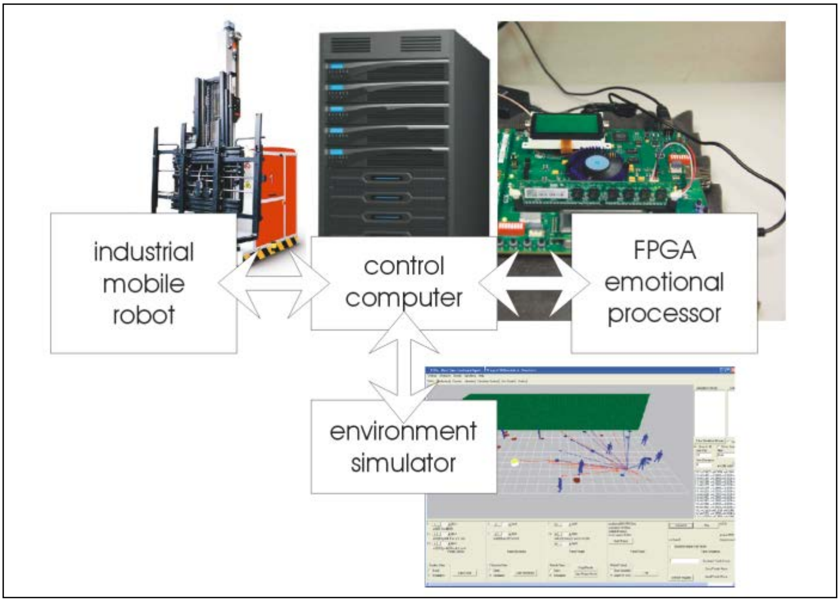

Fig. 3.9 FPGA Stratix III-based robotic agent architecture

To define the emotional computational workload of the applications, a simulator of the agent environment is used (see Fig. 3.10). This simulator allows defining different scenarios where the aforementioned activities are tackled (e.g., surveillance).

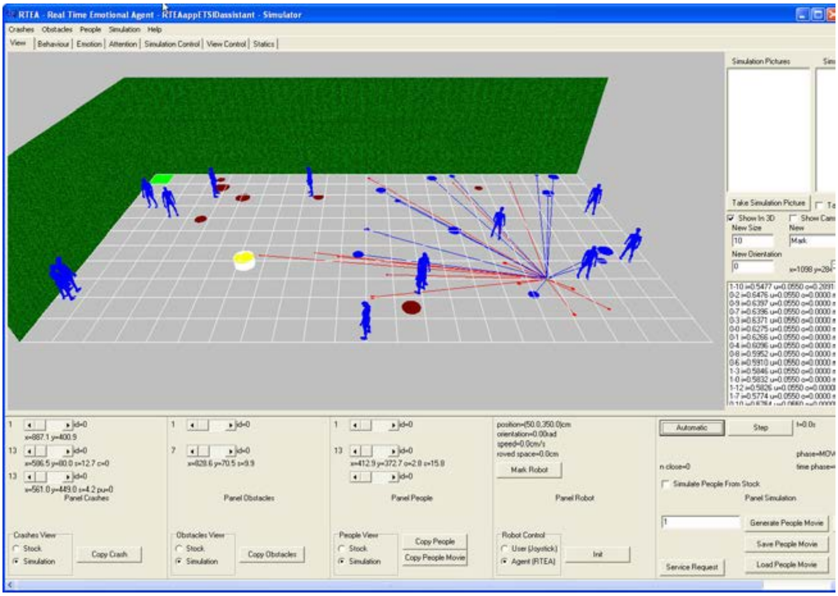

Fig. 3.10 Agent solving a crash in the operating area

The robot speed, the number of objects in the operation area, and the collision risk factors are used to define the attention cycle of the robotic agent, Ta, which is defined in the range of [0.1s , 0.5s]. Table 3.1 shows the robotic agent speed values used in the experiments.

Table 3.1 Robot speed

Safe	Normal	Risky
0.1 m/s	1 m/s	2 m/s

Table 3.2 shows the values of the complexity of three types of applications, measured in millions of emotion operations per attention

cycle (Mopc). To obtain these values, applications of different complexities are run in the simulator environment and the number of emotions involved in each of the applications is calculated. A simple problem requires the execution of about 0.5Mopc, while a complex application involves the execution of 2Mopc.

Table 3.2 Complexity of application problems

Number of Emotional Operations per Attention Cycle		
Simple	Normal	Complex
0.5Mopc	1Mopc	2Mopc

The emotional state of the robotic agent represents the ratio between the time spent to execute the emotional processes and the time of the attention cycle as shown in Equation 3.7.

$$\varepsilon = t_e / T_a \tag{3.7}$$

Where, t_e is the processing time of the emotions.

Three robotic agent emotional states are considered in the experiments (i.e., relaxed, normal and stressed). In the ideal situation, the emotional computational time, t_e, in the relaxed mode is less than 10% of T_a, in the normal mode it is between 10% and 25%, and in the stressed mode it is between 25% and 40% as shown in Fig. 3.11. A workload higher than 40% is not be acceptable because the process applications are stalled and the robotic agent cannot fulfill the objectives.

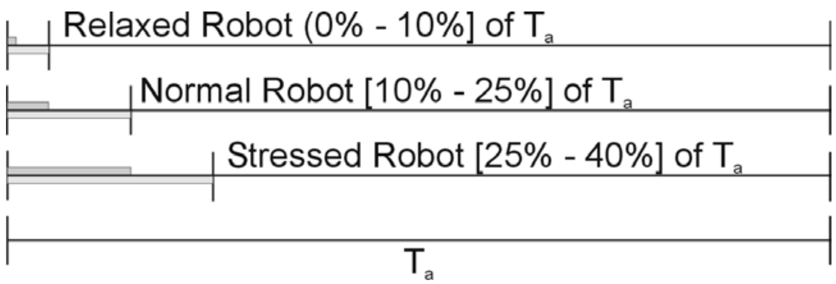

Fig. 3.11 Emotional workload limits

In the relaxed mode, the robot dedicates less time to the emotional processing and more time to solve application problems. That is, using the FPGA, this device will have less than 10% of T_a to compute the emotional processes. In the opposite, for the stressed mode, the robotic agent dedicates more time to the emotional processing and hence the FPGA will have between 25% and 40% of T_a to process the emotions. This means that the throughput of the FPGA in the relaxed mode is much higher than in the stressed mode. Table 3.3 shows the three emotional states used in the experiments, when the robotic agent undertakes the resolution of the different problems.

Table 3.3 Agent global emotional state ε

Relaxed	Normal	Stressed
< 0.1	[0.1, 0.25]	[0.25, 0.4]

The environment simulator (see Fig. 3.10) has been programmed with different robotic applications where the different combinations of the complexities of the problems, the robot speeds, and the emotional states are applied in order to calculate the emotional computational costs. Table 3.4 summarizes the obtained emotional costs, measured in MOPS, for each of the robotic scenarios.

Table 3.4 Emotional processing cost for robotic agent applications (MOPS)

Robot speed				
Problem	**Robotic agent state**	***0.1 m/s***	***1m/s***	***2m/s***
Simple	Stressed	3	5	11
	Normal	4	8	17
	Relaxed	13	25	51
Normal	Stressed	6	11	19
	Normal	8	17	33
	Relaxed	26	49	99
Complex	Stressed	9	21	39
	Normal	17	33	68
	Relaxed	51	99	200

The emotional computational requirements shown in Table 3.4 have to be fulfilled by the FPGA processor in order to allow tackling the corresponding type of problem. For instance, a simple problem using a relaxed robot running at the maximum speed (see Table 3.4) will require an FPGA that can process at least 51MOPS. If the FPGA is not able to support this throughput, the robot will fail to solve this problem.

3.4.2. FPGA-based emotional system design

In this section, the implementation of the emotional system presented in section 3.3 is developed using Stratix III and IV FPGA's.

3.4.2.1. Emotional processor design

Fig. 3.12 shows a block diagram of the FPGA based agent control system. The single-core executes the application processes of the behavior system, while the FPGA implements the operational processes of the emotional system.

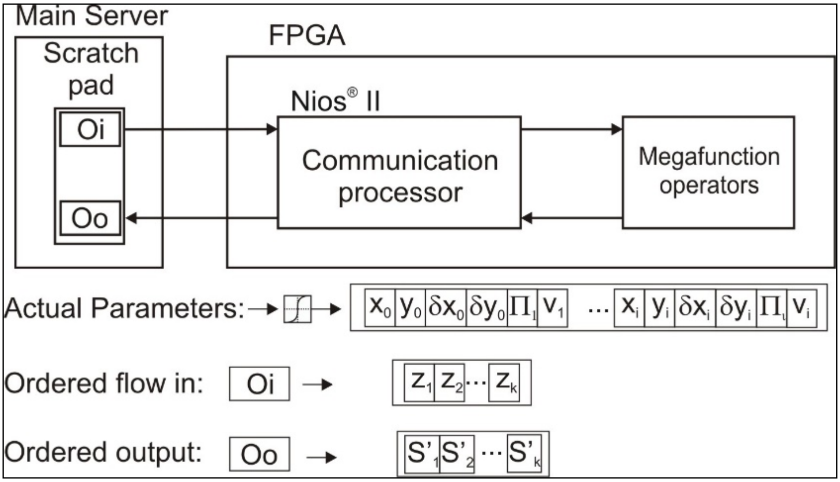

Fig. 3.12 Emotional processor

The Megafunction operators block implements the emotional processes and contribution functions. The Nios II is a software processor (IP by Altera) used to communicate the FPGA emotional processor with the main server of the robotic agent controller. Operators of the Altera Megafunction floating point library work synchronously and are implemented on a segmented basis. This makes it possible to process a continuous stream of data (pipeline) at the operating frequency of the FPGA device. The latency of each operator is due to its internal structure and the algorithm in which it is based. FIFO buffers are used to synchronize the operations according to their latencies.

The proposed design is based on A3 methodology to exploit the parallelism of the emotional processes so as to find the potential factorization with the aim of using the minimum number of operators to process the maximum number of operations. The emotional system is based on Dot-Product functions (contributions), which are applied to the appraisal data. Since emotions can have a variable number of contributions, the emotional processor is designed around an emotion operator, which processes a sequence of contributions and adds them to the emotional state.

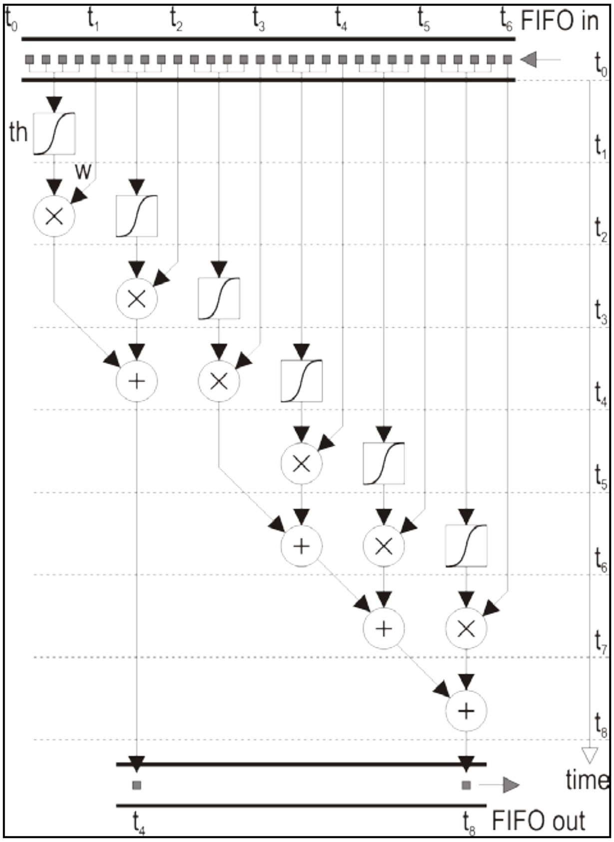

Fig. 3.13 Data flow graph for two emotions

The Data Flow Graph of an emotion operator is shown in Fig. 3.13, where an example of the execution sequence of two emotions is shown. The data input is performed through the FIFO-in and the emotional states are obtained through the FIFO-out. The flow rate of the obtained emotional states at the output is lower than the flow rate of the input parameters (6 parameters * number of emotional contributions). The first emotion, with two contributions, completes its evaluation at t4. The second one has 4 contributions and finishes at t8. Emotional contributions are based on hyperbolic tangent functions (th) and each contribution has a weight (w). Although Fig. 3.13 shows the temporal separation between the processing of each contribution on the same operator in a simplified way, in the pipeline each operator processes several contributions overlapped in time (as many as the size of the pipeline).

The pipelined-processing of the emotional contributions can reach a frequency in between 200MHz and 300MHz depending on the design synthesized on the FPGA. However, the accumulation phase of the contributions in an emotional state must be performed with a feedback adder, so that the processing rate must be reduced depending on the latency of the addition operator (14 cycles). To avoid this latency, the main processor, when it sends the data to the emotional processor, it interleaves the contribution parameters of groups of emotions (14 emotions) in order to accommodate the data flow to the latency of the accumulation phase.

Using ordered IN and OUT data flows a performance of 265 million of emotional contributions per second (MOPS) can be achieved. In addition to the pipelined processing, the emotion operator has been also replicated to process in parallel. FPGA devices used in the experiment have sufficient resources to implement multiple emotion operator replicas (7 in Stratix III and 8 in Stratix IV). In this case, multiplexers are used to distribute the data to the emotion operator replicas as shown in Fig. 3.14.

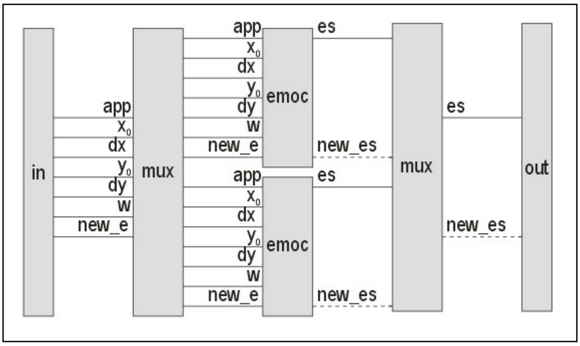

Fig. 3.14 Two replicas of the emotion operator

The input stream receives the parameters: x_0, *dx*, y_0, *dy*, *w*, *app* (appraisal) and *new_e* (start new emotion), then the multiplexers send the parameters to the emotion operators. Finally, the output stream sends the emotional state, *es*, and the *new_es* signal, indicating its availability.

To implement the emotion, a modular design is followed. First, the hyperbolic tangent is implemented using the available megafunctions. The latency of this function is 56 clock cycles. Fig. 3.15 shows the implementation of the parameterized hyperbolic tangent. FIFO queues are used to adjust the latencies and synchronize the parameters. The total latency of this process is 106 clock cycles.

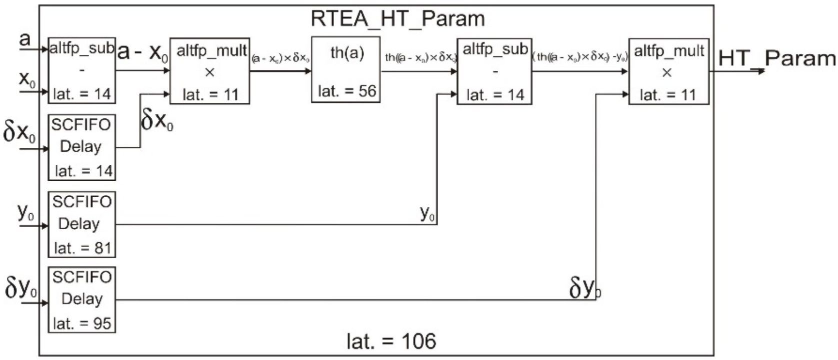

Fig. 3.15 Parameterized hyperbolic tangent

The parameterized hyperbolic tangent is multiplied by the weight *w* to obtain an emotional contribution; this process has a latency of 117 clock cycles. Finally, an accumulation phase completes the emotion operator with a latency of 131 clock cycles (see Fig. 3.16).

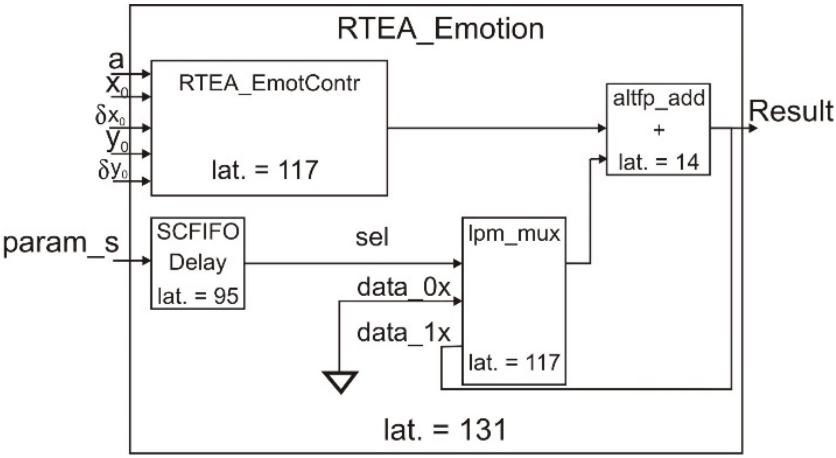

Fig. 3.16 Emotion operator

The emotion operator (see Fig. 3.16) is replicated in the selected FPGAs (Stratix III and IV) to process emotions in parallel. This replication will allow increase the throughput to calculate the millions of emotions composing the emotional system.

3.4.2.2. Design synthesis of the emotional system

The emotional system is simulated on Stratix III and Stratix IV, and synthesized on Stratix III. The EP3SL150F1152C2N device is implemented on a Stratix III EP3SL150N development board, which is the available system in our laboratory. Likewise, the processor is simulated on a Stratix IV, implementing the EP4SGX230KF40C2 device, since it is a FPGA family widely available in the industry. The type of resource that has limited the design is the number of integrated DSPs (Stratix III: 112, Stratix IV: 161).

Quartus II software, which allows structural design, is used together with Megafunction library. This library defines arithmetic operators for real floating point numbers IEEE-754, with simple (32-bits) or double (64-bits) precisions. The developer can define other formats, since the exponent and mantissa fields are configurable. Instead of codifying the basic processing resources directly in VHDL the Altera Megafunction library has been chosen because its functions are optimized for Altera devices and most of them let the developer to choose the optimization criteria (speed or area). In the proposed design, floating point real numbers are used as simple precision. Integer or fixed point real number representations were discarded in front of the floating point, because this representation has the following advantages for our design:

(i) provides a direct interface with the processes in the main processor avoiding the time spent in the process of transformation of the real numbers represented in main processor and FPGA.

(ii) facilitates the performance analysis between FPGA and Multicore since the same representation in both systems is used.

(iii) and allows flexibility to adjust the bit-widths of the operators which allows us to modify the range and accuracy of the values to suit different types of emotional sensitivity. These are important features of the research, and the use of this representation is not an excessive penalty, as we have enough resources in the considered Stratix FPGA devices.

The Megafunctions listed in Table 3.5 are used, and the option to optimize the area is selected. The latency, when there are different function versions is set to maximize the clock frequency allowed for a specific function. Table 3.5 shows the parameters for the Stratix III device (ALUT – Adaptive Look-Up Table, DLR – Dedicated Logic Register, ALM – Adaptive Logic Module and DSP is an 18-bit DSP).

Table 3.5 Megafunctions used in the emotional processor design

ALTFP Function	**Latency (clock cycles)**	**f_{MAX} (MHz)**	**ALUT**	**DLR**	**ALM**	**DSP**
ADD_SUB	14 (high latency)	416	599	603	427	-
DIV	14 (low)	296	295	331	262	16
MULT	11 (high)	466	195	301	206	4
EXP	17 (low)	275	631	521	445	19

Table 3.6 shows the resource utilizations of Stratix III and Stratix IV, summarized for the designed process emotion block. The values of the Stratix IV are represented in brackets. The utilization is represented as a percentage of the available resources, which permits to have an initial estimation of the number of replicas of every synthesizable emotional operator on each device.

Table 3.6 Resources utilization using Stratix III and Stratix IV

Type of Resources	**Total Resources**	**Emotion** 5% (4%)
Combinational ALUTs	*113,600 (182,400)*	4,091-4% (3,699 - 2%)
Memory ALUTs	*56,800 (91,200)*	662-1% (0 - 0%)
Dedicated Logic Registers	*113,600 (182,400)*	5,640-5% (6,867 - 4%)
Total Block Memory Bits	*5,630,976 (14,625,792)*	17,536 -< 1% (17,536 -< 1%)
DSP Block 18-bit Elements	*384 (1288)*	51-13% (51-4%)

Table 3.7 shows the maximum operation clock frequency for the emotional processor depending on the selected device. TimeQuest Static Timing Analyzer is used to obtain these values.

Table 3.7 FPGA frequency

Device	**clock - F_{Max} (MHz)**
Stratix III	265.67
Stratix IV	311.53

The maximum frequency, Fmax, is shown for a single instance of the emotional process. So, it is possible to maintain an in-flow (situation appraisals) at Fmax frequency, then the emotional processor would be able to evaluate 265MOPS and 311MOPS, respectively. This means that after a latency of 131 clock cycles, a constant output flow (emotional states) at 265MOPS or 311MOPS, depending on the device, can be obtained.

Table 3.8 shows the effect on the Fmax when the emotion operator is replicated into the emotional processor. This table shows the maximum number of replicas implemented in the FPGA devices due to the limitation of their available resources (the number of DSPs in the proposed design).

Table 3.8 Fmax with emotion replication

Device	Number of replicas	clock - FMax (MHz)
Stratix III	4	253.42 (x0.95)
	7	209.64 (x0.79)
Stratix IV	4	269.54 (x0.86)
	8	230.47 (x0.74)

When dealing with the emotion operator replication in the FPGA, a key point is the bottleneck of the communication between the FPGA and the single-core. In the proposed design, 24 bytes (six simple floating point parameters x_{0i}, y_{0i}, δ_{xi}, δ_{yi}, w_i, a_i) per contribution function are sent over the input flow. In a complex problem (highest speed and relaxed robotic agent) where 2M emotional contributions are executed per attention cycle, the number of bytes transmitted is 24 bytes x 200 Millions functions/s, which means 38.4GB/s. To transmit this data flow, a PCIe 3.0 interface at 40Gb/s is suitable.

Table 3.9 shows the performance of the communication IP cores currently available for Altera FPGA devices. The selected communication interfaces of each device are selected to maximize the throughput: Ethernet for Stratix III and PCIexpress for Stratix IV. As shown in Table 3.9, the performance of the emotional processor in this case is limited to 40MOPS using Stratix III, and to 208MOPS with Stratix IV.

Table 3.9 FPGA performance for the available IP communication cores

Bus performance	(Gb/s)	Processor performance	(Mops)
PCIe 3.0	40	Stratix IV	208
USB 3.0	4.8	S. III-E, S. IV-GX, S. IV-GT	25
HyperTransport 3.1	10	S. IV	40
Ethernet	10	S. III, S. IV	40
	40	S. IV-GT	160

3.4.3. Partitioned multicore-based emotional system design

Regarding the implementation on a Multicore, the robotic agent architecture including the belief, behavior, attention and emotional sub-systems is implemented as a partitioned system [22] on a six-Core processor; the Intel Core i7-980X at 3.33GHz per core. The i7 based computer has 8MB cache memory, 12 GB DDR3 RAM. One of the cores is dedicated to the attention system, a second core is used for the behavior-belief systems and the remaining 4 cores implement the emotional system. The number of active cores for the emotional system can be configured by the operating system, using the *sched_setaffinity()* system call in Linux. For evaluations purposes, the emotional processor is run as Single-core mode, Dual-core and Quad-core in order to assess which is the sufficient number of cores to tackle the robotic applications. In the Single-core mode, all the emotional processes are assigned to one processor. In the Dual and Quad core modes, the processes are distributed evenly among the different cores. To this end, the Worst Fit algorithm is used to refill the cores [20]. The system model implemented is shown in Fig. 3.17.

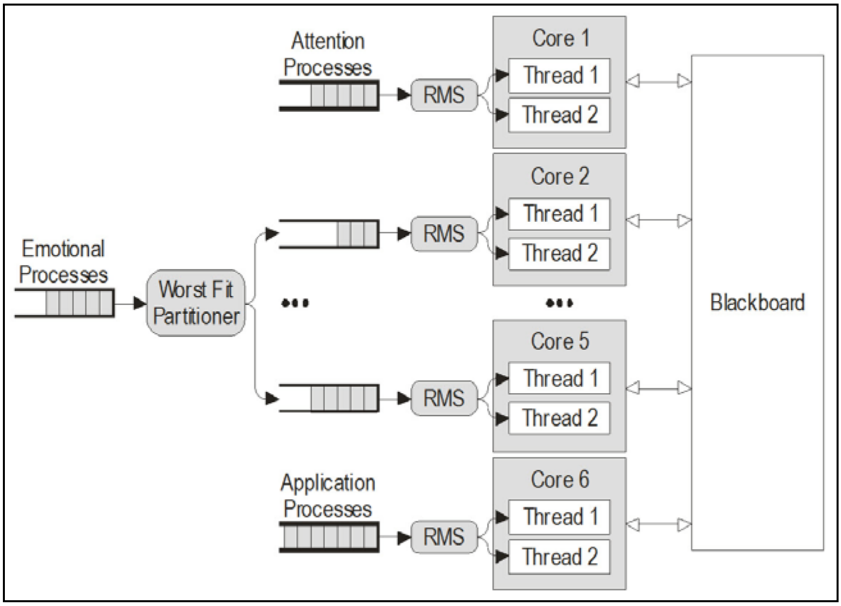

Fig. 3.17 Multicore architecture

Initially, the WF replenishes the core with the emotional processes. The policy of this strategy is to assign the process to the less loaded core until all the processes are assigned to their corresponding core. Each core implements a Rate Monotonic Scheduler to execute the processes belonging to that core [21]. That is, the process with the shortest period has the highest priority for execution in the core. The schedulers and the processes are implemented as real-time tasks in the rt-linux kernel. The real-time processes are executed using the main memory without accessing the disk device.

The cores share a memory structure called blackboard [23]. Through this memory the processes read and write the important data as the appraisals, emotional states and motivations. Also it allows, using shared variables to synchronize the different processes. The attention core, at each attention cycle, activates the Belief processes that write in the blackboard the appraisal information. The emotional processes then read the appraisals and calculate the emotions to update the motivations in the memory. Finally, the Behavior processes read the motivations and execute the actions prioritizing the behaviors with highest motivations. These operations are repeated periodically at each attention cycle.

3.5. Experimental evaluation

In this section, the implementations of the emotional architecture in FPGAs Altera Stratix III, IV and Multicore processors are compared. The comparison is performed by executing on each of the platforms different complexity levels applications of the robotic agent. The evaluation is focused on the analysis of the performance, measured on MOPS, that FPGAs Stratix III and IV, and multicore provide to solve the agent problems.

Fig. 3.18 to Fig. 3.26 show the results of evaluating the different implementation alternatives, considering 3 complexity problems (Simple, Normal and Complex) and 3 robotic agent emotional state levels (Stressed, Normal and Relaxed) for 3 different values of the robot speeds (0.1m/s, 1m/s and 2m/s). In each figure, the bars represent the maximum computation capacity in MOPS that each processor or FPGA can provide (e.g., Fig. 3.18 shows that a Stratix IV allows 208MOPS, a Quad-core performs 91MOPS while a Dual-Core reaches 46MOPS).

For each pair (complexity problem, robotic agent emotional state), the speed of the agent limits the computational capacity (horizontal lines) required for each processor to solve a specific problem. For instance, in the case of a (simple problem, relaxed robot), if the speed is 2m/s the minimum required computation capacity to solve the problem is 51MOPS, while at 0.1m/s it is 13MOPS (see Fig. 3.18). This is because the attention cycle increases as the speed is reduced and hence more time is available to solve the same problem, then less computation MOPS are required. At the former speed, only Quad-Core and Stratix IV can solve this kind of problems, while at the latter speed all the considered processors can tackle the problem. In general, for the same type of problems, at higher speeds, the computational requirements of the processors increase.

On the other hand, as the complexity of the problem increases, the processor computation requirements also increase. For instance, for a normal problem and relaxed robot at 1m/s the required MOPS are 49 (see Fig. 3.21), while a complex problem with the same relaxed robot at the same speed, requires 99MOPS (see Fig. 3.24). Moreover, for the same kind of problems, if the robot emotional state is becoming more stressed, then the FPGA/Multicore computational requirements decrease because the execution time of the emotions is higher, and hence less computing power is required from the FPGA-Multicore processors. For instance, Fig. 3.24 shows that for a complex problem and relaxed robot at 2m/s, 200MOPS are required, while Fig 3.26 shows that the same problem with a stressed robot at the same speed requires only 39MOPS. This is due to the fact that if a relaxed robot is desired then more throughput is required from the FPGA/Multicore.

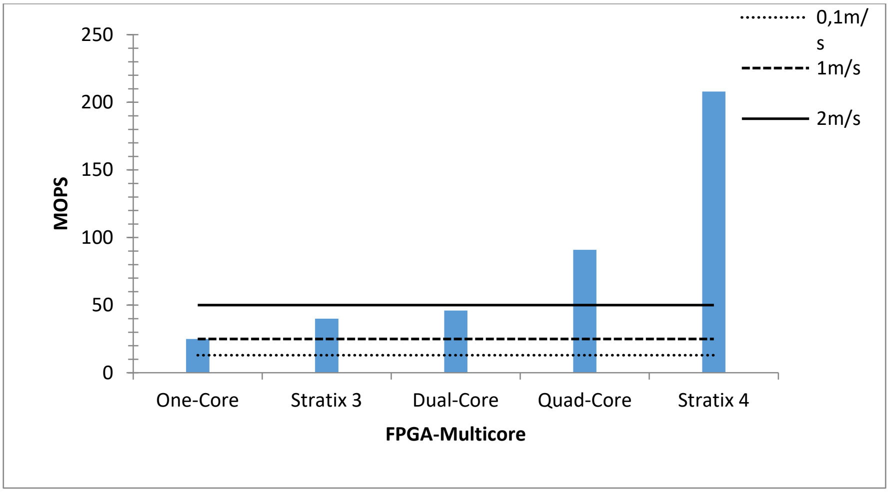

Fig. 3.18 Simple problem – relaxed robotic agent

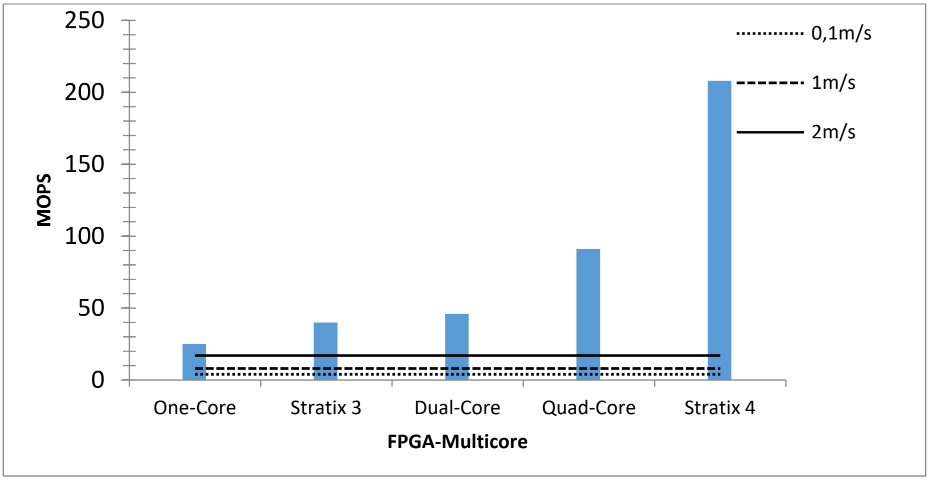

Fig. 3.19 Simple problem – normal robotic agent

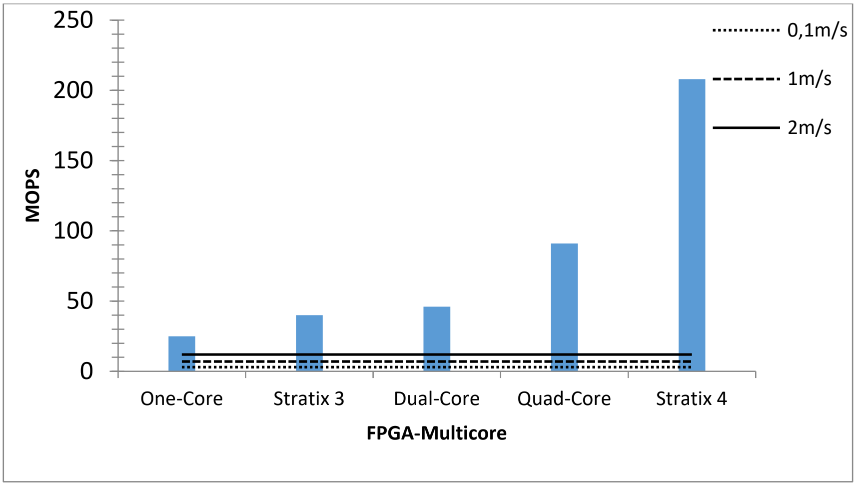

Fig. 3.20 Simple problem – stressed robotic agent

Analyzing the results in more details, it is noted that all the platforms (single-core, Stratix III, Dual-Core, Quad-Core, Stratix IV) can solve simple problems with a stressed or normal agent at any speed. However, with a relaxed agent, single-core can only support the application if the agent runs at the lowest speed (0.1m/s), Stratix III and Dual-Core solve the problem at low and medium speeds, while Stratix IV can afford it even at the maximum speed.

For the normal defined problems, when the agent is stressed all the evaluated processors can solve the applications at any speed. Using a normal agent, only the single-core fails to solve the problems at the maximum speed. A relaxed agent can solve the applications by using Stratix III, Dual-Core, Quad-Core and Stratix IV at 0.1m/s. If the speed is increased at 1m/s only Quad-Core and Stratix 4 can solve the problems. Finally, at 2m/s only Stratix IV can tackle the situation.

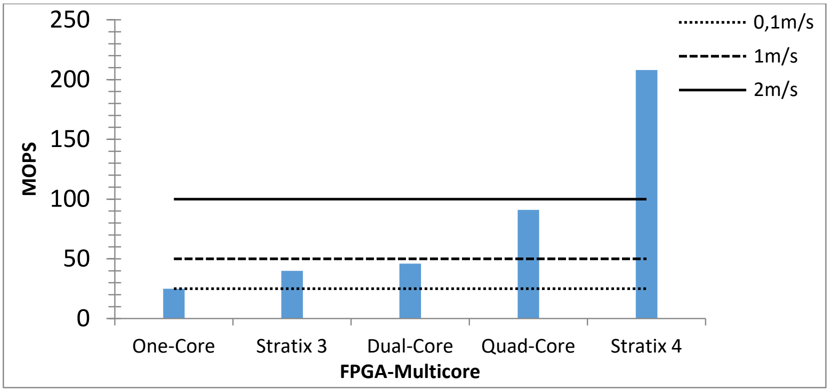

Fig. 3.21 Normal problem – relaxed robotic agent

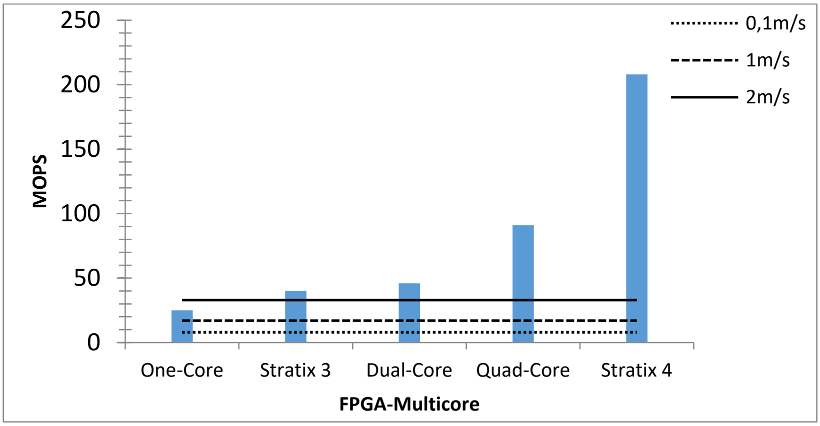

Fig. 3.22 Normal problem – normal robotic agent

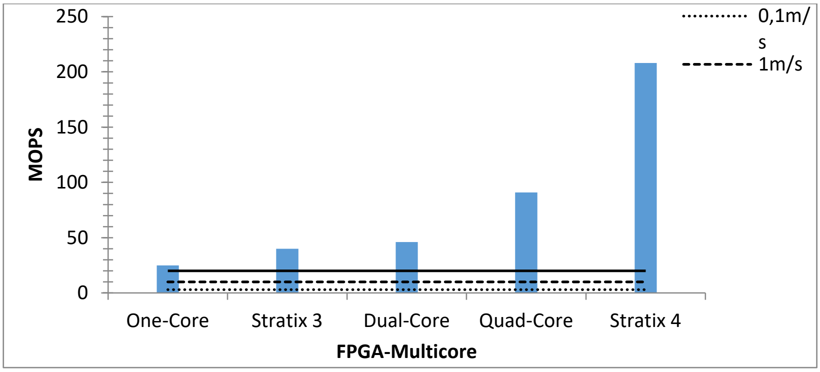

Fig. 3.23 Normal problem – stressed robotic agent

Regarding complex problems, using a stressed agent at low and medium speeds, all solutions solve the applications. At the maximum speed, only Dual-core, Quad-core and Stratix IV are applicable. For a normal agent, at the minimum speed all solutions are valid. At 1m/s a single-core cannot solve the situation. Increasing the speed at 2m/s only Quad-Core and Stratix IV can solve the applications. If the agent is relaxed, at the minimum speed, only Quad-core and Stratix IV are adequate solutions. Increasing the speed more than 1m/s causes that only Stratix IV is a valid solution to solve the application.

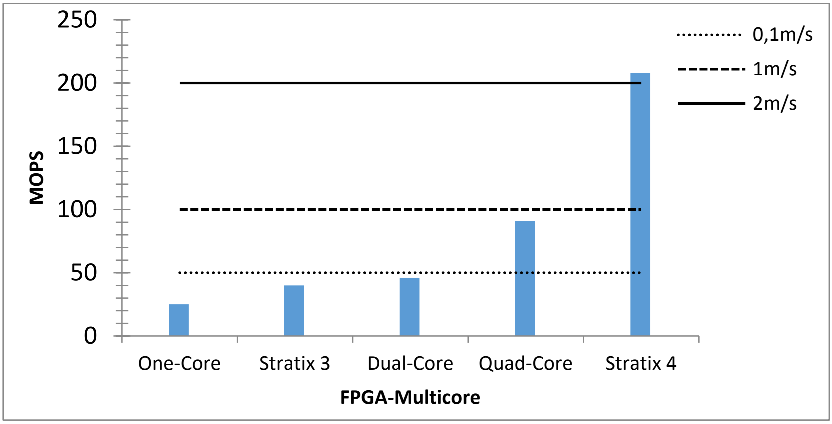

Fig. 3.24 Complex problem – relaxed robotic agent

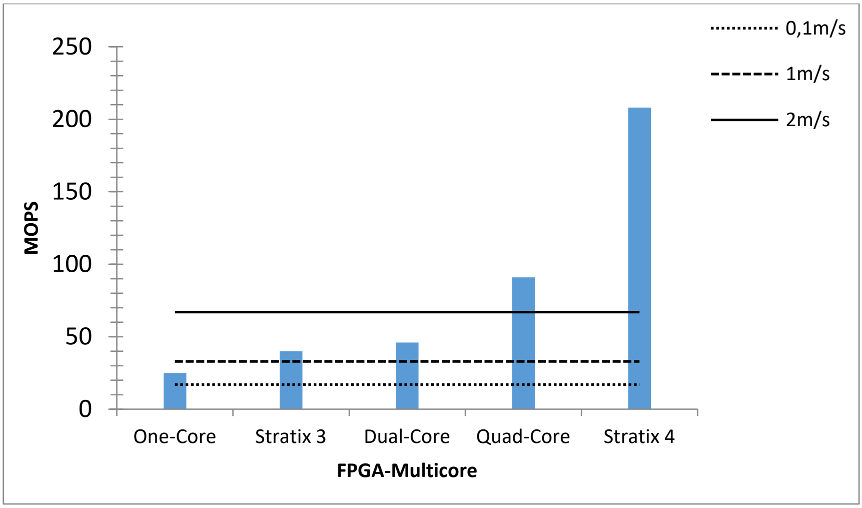

Fig. 3.25 Complex problem – normal robotic agent

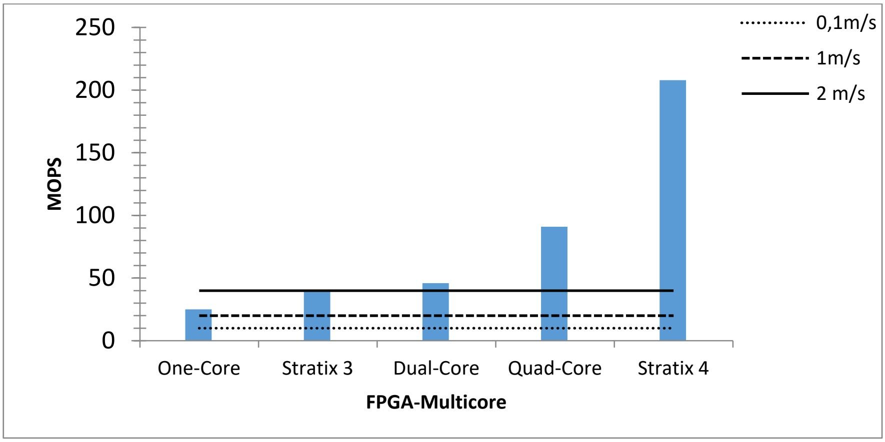

Fig. 3.26 Complex problem – stressed robotic agent

As a summary of the evaluation, it can be pointed out that the obtained performance of the Stratix IV implementation of the emotional processor increases the performance of the initial implementation of the architecture in about one order of magnitude. As a consequence, all the complex applications that could never be executed using the initial version using a single-core (max. 25MOPS) now they can be undertaken (e.g., complex problem and relaxed robot - min. 50MOPS).

The study shows also that other less expensive solutions using Stratix III could be applied to solve problems (e.g., complex problem and normal agent at 1m/s -38MOPS) with around the double of the requirements that a single-core could support. Dual-Core provides slightly better performance than Stratix III, so it can be used to solve some problems that Stratix III cannot solve, such as the complex problem and normal agent at 1m/s (33MOPS). Moreover, Dual-core is relatively cheaper so it is a better choice than Stratix III.

Using Quad-Core, the performance of the architecture is increased in 3.64 times in relation to the first implementation. Thus, from the 27 proposed applications about 89% can be solved. However, using the original implementation only 55% can be tackled. Furthermore, Quad-Core has a lower cost than a Stratix IV, so more adequate solution but only if the type of applications to carry out is not the most complex one.

3.6. Conclusions

An FPGA based emotional control architecture to implement future robotic agents has been presented. The emotional processes of the architecture have high computational requirements, which consumes the computational power of the main processor. To reduce this consumption, the parallel capabilities of the emotional processes of the architecture have been exploited and the implementation of the emotional processes on high performance FPGA processors has been tackled. An industrial mobile robotic agent application (under different environmental, dynamic and emotional robot state conditions), implementing the emotional based FPGA architecture has been proposed. The performances have been evaluated for FPGAs Altera Stratix III and IV, and the results are compared with the implemented emotional system in a Single-Core, Dual-Core and Quad-Core. Results show that Stratix IV implementation of the emotional processor increases the performance of the initial implementation of the architecture in about one order of magnitude. Stratix III could be applied to solve problems with around the double of the requirements that a single-core could support. Dual-Core provides slightly better performance than Stratix III and it is relatively cheaper so it is a better choice than Stratix III. Using Quad-Core, the performance of the architecture is increased in 3.64 times in relation to the first implementation. Thus, about 89% of the proposed applications can be solved. Quad-Core has a lower cost than a Stratix IV, so more adequate solution but only if the type of applications to carry out is not the most complex one.

Acknowledgements

This work was supported in part under Spanish Grant PAID/2012/325 of "Programa de Apoyo a la Investigación y Desarrollo. Proyectos Multidisciplinares", Universitat Politècnica de València, Spain.

4. Publicación B:

Embedded GPU and Multicore Processors for Emotional-based Mobile Robotic Agents

Francisco Almenar, Carlos Domínguez, Juan-Miguel Martínez, Houcine Hassan, Pedro López.

Future Generation Computer Systems. 56(March 2016) 192-201

DOI: 10.1016/j.future.2015.05.010

Abstract

Future robotic systems are being implemented using emotional architectures. In these architectures, the emotions have the central role of regulating the behaviors of the robots. The computation requirements of the emotions raise with the complexity of the problems, which limit the capacity of the processor to tackle them. Fortunately, the potential parallelism of emotional processes permits their execution in parallel, hence enabling the power computing to tackle the complex dynamic problems. In this paper, Graphic Processing Unit (GPU), multicore processors and single instruction multiple data (SIMD) instructions are used to provide parallelism for the emotional processes. Different GPUs, multicore processors and SIMD instruction sets are compared to analyze their suitability to cope with robotic applications. The applications are set-up taking into account different environmental conditions, robot dynamics and emotional states. Experimental results show that, despite the fact that GPUs have a bottleneck in the data transmission between the host and the device, medium and high performance GPUs permit undertaking complex robotic problems, while low performance GPUs allows solving medium and low size problems. In addition, although

SIMD instructions alone are not enough to undertake complex and some medium robotic problems, they allow obtaining some speed-up at zero cost, just by using processor intrinsic instructions. Dual-core processors show a similar performance to SIMD instructions, while the use of quad-core processors provide similar results as low performance GPUs.

4.1. Introduction

Many research works [1, 2, 3, 4, 5] predict a growth of the number of intelligent robots in the industry and in our lives in the two next decades. They state that advanced robots capable of making decisions on their own as humans do are still under development and the first prototypes will not start to appear until 2030. Some researches [2, 6] state that we are seeing the emergence of the first generation robots such as the demining robot Warrior manufactured by iRobot [7], which are able to solve simple tasks with little ability to adapt to the changing environment, and running their program code on a single-core processor. However, more intelligent features that robots could include such as decision-making are not yet developed in real robots. It is expected that, by 2050, robots will be implemented using advanced computers capable of running hundreds of billions of instructions (i.e., 4th. generation robots). These robots would rival human intelligence and would be able to perform operations of abstraction and generalization, medical diagnostics, planning and decision-making [3, 4, 5].

Control architectures based on emotions are inspired on emotional natural agents. They are becoming promising solutions for the implementation of advanced robotic systems [3, 8, 28, 9, 10] because they facilitate the process of decision-making [1, 18]. They use the mechanism of emotion in organizing the behaviors, which has the following advantages: allow the robot to be autonomous to focus its attention on the most promising behavior; provide a bounded response time, which helps organizing the deliberative processes; sort the problems based on the expectations of success; autonomously adapt the computational load to the available processor capacity allowing solving problems of increasingly complexity; separate the decision from the action processes; and use of subjective appraisal of the situation permit finding always an alternative solution. In this paper, an emotional robotic architecture for the control of complex mobile robot applications is used. In this model, two main types of processes coexist: behavior and emotion processes. The former solve the application problems (e.g., surveillance) while the latter use an emotional mechanism to motivate the robot behaviors.

Originally, all the processes of the emotional architecture, including the behaviors and the emotions were executed on a single-core computer (e.g., Intel 2,6GHz). The emotional processes must be applied to all problems/subproblems of the robot agenda at every cycle of attention (e.g., 0.1s). As the agenda grows in high complexity level applications, the emotional workload increases significantly as well (e.g., 200 million operations per second (MOPS)).

Each one of these operations is a reduction function involving: an hyperbolic tangent function, a multiplication and a sum of up to 6 other functions. However, the control computer did not support this intensive workload because it could only execute up to 25 MOPS. Moreover, the implementation of the emotion processes on an MCU or low to medium performance DSP was discarded because these devices provided even less power computation (i.e., between 10 and 20 MOPS). Alternatively, we can use FPGAs to provide the processing capacity problem (i.e., Stratix IV by Altera). In our preliminary experiments, Stratix IV [13] was able to solve even complex problems. However, they have a high cost [29].

Fortunately, there are some alternatives. Taking into account the inherent parallelism of the problem, this paper proposes the use of multicore processors, GPUs and SIMD instructions to implement the emotional system. All these alternatives do not need of any special hardware except from the ones that can be found in any modern computer. By executing the emotion processes with the proposed alternatives, the control computer will get slack time to solve more complex applications by: (i) improving system throughput by simultaneously executing several problems, or (ii) tackling more time critical dynamic problems (e.g., solve the problems at a higher speed).

The rest of the paper is organized as follows. Section 4.2 shows some related work. Section 4.3 describes the problem to be solved and the sequential algorithm that implements it. Section 4.4 describes the alternative parallel implementations explored in this paper. Section 4.5 present some evaluation results for different scenarios. Finally, some conclusions are drawn.

4.2. Related work

Control architectures inspired by the cognitive mechanism of the human mind are becoming promising solutions for developing advanced robots. One type of the cognitive architectures is based on emotions [1, 9, 10]. Different research groups [2, 6, 18] are focusing on the design of the control architectures for emotional-based robots. Salichs [18] proposes a decision making system based on drives and motivations, also based on emotions and auto-learning. The aim of the agent is to learn how to behave through the interaction with the environment, using reinforcement learning, to maximize their well being. Moshkina et al. [6] develop an algorithm based on the emotional disappointment of the robot. To achieve it, they get inspiration from the disappointment observed in animal and humans. Simulations show that robots which include this emotion are more effective than the traditional ones. Damiano [2] suggests a model where the decision making is based on a motivational system. Motivations have a value that depends on the necessity that has to be satisfied and incentives stimulates.

Once all the values are calculated, the biggest motivation activates and organizes the behavior trying to satisfy the most urgent necessity. Lee-Johnson et al. [8] develop a hybrid architecture reactive/deliberative that incorporates artificial emotions to improve the decision making and the actions of a mobile robot. These emotions are active on different levels of the architecture, they modulate decisions and actions of the robot. Moshkina proposes an effective model called TAME [6] to help with the interaction between the man and the machine. It is based on different concepts like the emotional state, the emotion and the attitude. These works propose interesting models based on emotions however, implementations of these models are usually done with sequential algorithms using general-purpose processors, and consequently increasing the cost.

The aim of this paper is to parallelize these emotional processes to improve performance at a reasonable cost. High- and low-performance GPUs, multicores and SIMD instructions are used to parallelize these processes.

There are some works related to the implementation of emotional systems using high performance hardware. In [30], authors propose an implementation of an emotion bio-inspired system. In this work, the authors design a FPGA controller based on emotional learning. However, the application consists of the control of a simple crane, which could be solved using a traditional PID controller.

In [29], different possibilities to parallelize a limited subset of motivational processes and its implementation using a Stratrix IV FPGA are proposed.

The results obtained improve the implementation of the system in a single-core processor. However, the cost of migrating all the operating processes of the robot to the FPGA resulted in a quite prohibitive solution. Ducrot et al. [31] present a map estimating process with 2 depths and a partial implementation using GPU processors. They use a configuration of the Cuda-Core 448 architecture combined with dual-core processors. Their purpose is constrained to just static objects. In [32] authors presented an implementation of the R* search algorithm applied to complex planning problems, and fulfilled to reducing the cost of implementation. They propose to apply this solution to a real robot in future works. However, works where GPUs, Multicores and SIMD instructions are applied to speed up the processes that implement emotional robotic models to reduce the cost of the implementation of the 4^{th} generation robots, like this paper proposes, were not found in the bibliography.

4.3. Emotional architecture

4.3.1. Emotional processes specification

An emotion is the process of appraising an observed situation and motivating a robot behavior to undertake this situation. Fig. 4.1 details the emotional motivation process: (i) the emotional activation, and (ii) the emotional response.

The emotional activation sets an emotional state and the emotional response builds and motivates a behavior.

(i) During the emotional activation, the observed situations (1), represented as real properties, are subjectively appraised. The appraisal process depends on the robot character. The character dynamically adjusts the parameters of this process. To calculate the appraisal of the situation (5) the robot ponders and adds (4) a set of appraisal contributions (3), which are evaluated using contribution functions (2). Equation 4.1 represents the i^{th} situation appraisal.

$$a_i = \sum_{k=1}^{l} w_{ak} * f_{ak}(p_k) \qquad (4.1)$$

Where: p_k is the k_{th} property of the situation, f_{ak} is the k^{th} contribution function, w_{ak} is the weight of the function and l is the number of appraisal contribution in range of 1 to 6. The situation appraisals contribute to establish an emotional state (9). The emotional contributions (7), evaluated with contribution functions (6), are pondered and added (8) to finally give the emotional state. The emotional contributions functions are defined in the real range [-1,+1] and the emotional state in the [0, +1] range. Every emotional state is labeled in the robot navigation problem (e.g., fear of collision", a 0 level would mean no fear", while a 1 level would be afraid").

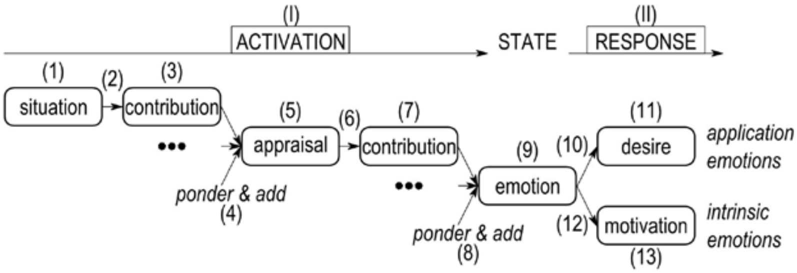

Fig. 4.1 Emotional motivation process

(ii) As an emotional response, the robot generates new desires (11), and motivates the behavior to accomplish the desire (13). The desires are the result of application emotions, and their response functions (10) depend on the problem, meanwhile the motivations are the result of intrinsic emotions, and their response functions (12) depend of the character of the robot. The j^{th} emotion is expressed as shown in Equation 4.2.

$$s_j = \sum_{i=1}^{l} w_{ci} * f_{ci}(a_i) \qquad (4.2)$$

Where: s_j is the state of the j^{th} emotion, f_{ci} is the i^{th} contribution function, w_{ci} is the i^{th} weight of the function.

The emotion contribution functions, f_{ci}, must have some properties such as slight variations at the ends of the range that tends to asymptotic values and abrupt variations around an inflection point in the center of the range. These properties are found in the hyperbolic tangent functions, which are used to represent the contribution functions:

$$f_{ci}(x) = th(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{4.3}$$

Where x is the appraisal value a_i when calculating the emotion. To allow adjusting the hyperbolic function, Equation 4.3 is transformed in the following function

$$th^*(x) = \left(\frac{e^{2(x-x_0)\delta_y} - 1}{e^{2(x-x_0)\delta_y} + 1} \right) * \delta_y \tag{4.4}$$

Where the parameters x_0, y_0, δ_x and δ_y permit to translate and scale the contribution function.

These emotions are grouped in the emotional system and have the structure shown in Fig. 4.2.

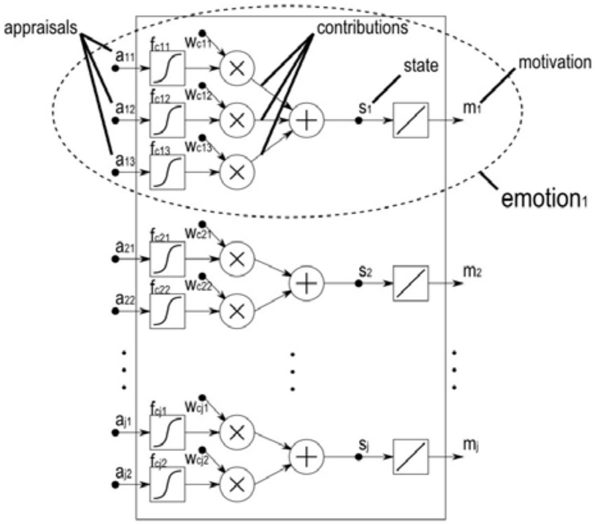

Fig. 4.2 Emotional system structure

The emotional system gets, at a given instant, inputs from a set of n situation appraisals (e.g., 2M, M refers to millions) and produces a set of K motivations (e.g., 0.5M). The hyperbolic tangent is applied to each appraisal and its result is multiplied by a weight. Each emotion can be composed of up to 6 different contribution functions. The obtained emotion can pass through a final function (f_i). In this paper, though, the identity function is used (i.e. no post-processing is done). The total number of these situation appraisals in the emotional system depends on the complexity of the problem, the environment conditions and the robot dynamics. In the experiments of the multi-objective robotic applications, this number reaches a value of about 200 MOPS. These operations can be computed in parallel since they are independent. However, in an initial implementation they were executed sequentially in a host computer controlling the robot. Due to their highly computational requirements, the capacity of the host processor was exceeded, being unable to fulfill the robot objectives.

In the next section, this paper exploits the inherent parallelism of the problem, proposing a parallel implementation that takes advantage of the hardware already available in modern computers to implement the emotional system.

4.3.2. Sequential algorithm

In this paper, the hyperbolic tangent is used to implement the emotional system of the robot (as shown in section 4.2.1). The following fragment of code shows an implementation in the C programming language of the calculation of the hyperbolic tangent. For the sake of simplicity, the translation and scaling factors and the weight of each contribution are not shown. This sequential code will be the basis for the parallel code that will be explained later.

```
for(i=0;i<n;i++) {
   fci[i] = (exp(2*a[i])-1)/(exp(2*a[i])+1); }
for (i=0;i<n/6;i++) {
   for(j=0; j<6; j++) {
      acum += fci[6*i+j]; }
   m[i]= acum; }
```

In the following section, different approaches to exploit the exhibited parallelism will be shown.

4.4. Parallel implementation of the emotional architecture

4.4.1. Multicore processors

The first parallel implementation of the emotional architecture is carried out using multicore processors. Multicore processors with several number of cores are standard in today's computers. The availability of several cores allows to execute in parallel several threads. To generate the parallel threads, OpenMP (OMP) is used. OMP is an API that supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran. This API modifies the run-time behavior to obtain thread-level parallelism. OMP relies on directives written by the programmer that tells the compiler what can be executed in parallel. Parallelism is obtained by forking a master thread into a specified number of slave threads; these slave threads receive a part of the task that the master thread has to perform, allowing threads to run concurrently. To indicate that a loop can be executed in parallel, the preprocessor directive #pragma omp parallel for is written just before the for loop. The code inside the loop does not need any special modifications. In this case, the counter of the inner loop (j) and the auxiliary reduction variable (acum) are declared as private. The code below shows the OMP parallel version of the emotional system:

```
#include <omp.h>

...

#pragma omp parallel for private( j, acum )

for( i=0; i<n/6; i++ ) {

  acum = 0;

  for( j=0; j<6; j++ ) {

    fci[6*i+j]  = (exp( 2*a[6*i+j])-1 ) / (exp( 2*a[6*i+j])+1 );

    Acum       += fci[6*i+j]; }

  m[i] = acum; }

   ...
```

4.4.2. Graphics processing unit

The second parallel implementation of the emotional architecture is based on the use of a GPU coprocessor. A GPU is a parallel, multithreaded many-core processor with a tremendous computational capability. This processing power is exploited by programmers by means of a programming model. Cuda is the programming model provided by nVidia, which is used in this paper. In this model, several blocks, each one composed of several threads, are launched to be executed in the streaming multiprocessor (SMs) available on the device [33]. The threads of a block execute concurrently on one SM In this paper, the workload of computing the n hyperbolic tangents is split among several blocks of threads. Before the GPU starts processing, input data should be allocated in the device where it will be processed. In addition, once the processing has finished, the resulting data has to be transferred back to the host computer main memory. Both data and results are transferred between the host and the GPU and vice-versa through the PCI express bus.

The time used to perform these transfers is added to the total execution time and its impact could be important. This is the case of the emotional architecture considered in this paper, where input data is only used once per computation. In this paper, it is assumed that every time that the robot needs to calculate its emotions to make a decision, the device memory has to receive all the input data, including the one which did not changed. So, the obtained results could be considered as pessimistic. One way to improve the results is to transfer only those data that have changed since the last run, therefore avoiding useless transfers. Another way of improvement is to overlap communications and computations [34]. Anyway, even without these improvements, the obtained performance of the GPU-based implementation outstands the rest of proposals.

The code below shows the Cuda-C implementation of the emotional architecture:

```
#include <cuda.h>

...

__global__ void hyperbolicTangent( float *dev_a,

                                   int   *dev_n,

                                   float *dev_fci ) {

  int tid = threadIdx.x + blockIdx.x * blockDim.x;
```

```
  while( tid < *dev_n ) {
     dev_fci[tid]  = tanh( dev_fci[tid] );
     tid          += blockDim.x * gridDim.x; }}

__global__ void reduction( float *dev_fci,
                           float *dev_m,
                           int *dev_n ) {
  int tid = 6 * threadIdx.x + blockIdx.x * blockDim.x;
  int tidemotion = threadIdx.x + blockIdx.x * blockDim.x;
  int count;

//Each thread adds the values of 6 contiguous
// hyperbolic tangents, to form an emotion,
  while( tid < *dev_n ) {
    count = 0;
    dev_m[tidemotion] = 0;
    while( count <= 5 ) {
      dev_m[tidemotion] += dev_fci[tid+count];
      count ++; }
    tid        += 6 * blockDim.x * gridDim.x;
    tidemotion +=     blockDim.x * gridDim.x; }}

int main(...) {
  float *a,     *fci,     *m;
  float *dev_a, *dev_fci, *dev_m;
...

//allocate memory on the CPU
  fci = (float*)malloc( n   * sizeof(float) );
  a   = (float*)malloc( n   * sizeof(float) );
  m   = (float*)malloc( n/6 * sizeof(float) );
```

```
//allocate memory on the GPU
  cudaMalloc( (void**)&dev_a,   n * sizeof(float));
  cudaMalloc( (void**)&dev_fci, n * sizeof(float));
  cudaMalloc( (void**)&dev_m, n/6 * sizeof(float));
  cudaMalloc( (void**)&dev_n,       sizeof(int));

//copy values from CPU to GPU
  cudaMemcpy( dev_a, a, n * sizeof(float),
              cudaMemcpyHostToDevice);
  cudaMemcpy( dev_n, &n, sizeof(int),
              cudaMemcpyHostToDevice);
  hyperbolicTangent<<<blocks, threads>>>( dev_a,
                                          dev_n,
                                          dev_fci);
  reduction<<<blocks, threads>>>( dev_fci,
                                  dev_m,
                                  dev_n );

//Copy emotions back from the GPU to the CPU
  cudaMemcpy(m, dev_m, ( N/6 ) * sizeof(float),
             cudaMemcpyDeviceToHost);
...

  cudaFree( dev_a );
  cudaFree( dev_fci );
  cudaFree( dev_m );
  cudaFree( dev_n );}
```

Before processing the data, we have to allocate it on the device's memory. To do so, `cudaMalloc((void**) &dev_a, n * sizeof(float))` is used. This function call works similar to the function `malloc` in C, and indicates that the vector `dev_a`, which has a size of `n` floats, is allocated on the GPU's memory. Once the memory is allocated on the device, we are able to transfer the data from the host memory. The `cudaMemcpy(dev_a, a, n * sizeof(float), cudaMemcpyHost-ToDevice)` function does the work, copies all the values of the vector a in `dev_a`.

Notice that the last argument indicates the direction of the transfer.

After the data is stored on the device, a function call can be done in order to start the execution on the device. According to Cuda-C syntax, the `hyperbolicTangent <<< blocks; threads >>> (dev_a, dev_fci)` launches the execution on the GPU that computes the hyperbolic tangent. This function uses `dev_a` (input data) and `dev_fci` (output data) as parameters. During the execution, the device needs to know in which part of the shared variable `dev_fci` data should be read and stored. To take care of this point, the integer `tid` is used. When the function ends, the results are still in the device, so a transfer of data should be done to send the results to the host. To perform this work, the `cudaMemcpy(fci, dev_fci, n*sizeof(float), cudaMemcpyDeviceToHost)` call is used. It works similarly to the function used before, but now the direction has changed. The `reduction <<< blocks; threads >>> (dev_fci, dev_m, dev_n)` launches the aggregation of the hyperbolic tangent in groups of 6. In both cases, there is a special parameter which is declared between <<<>>>, which indicates the amount of blocks and threads that is going to be used (see [35] to adjust this parameter to increase the efficiency).

Once the results are in the host, the memory allocated in the device should be released; as it is done in C, `cudaFree` call does this work.

4.4.3. SIMD instructions

Finally, we will propose the use of the SIMD (Single-Instruction Multiple- Data) instructions that are part of the ISA of processors since the MMX instruction set extensions were introduced by INTEL in 1999. For a better comparison, we have implemented SIMD version only in one core, even though it is possible to combine SIMD and multicore parallelism capabilities to obtain better results.

SIMD instructions allow exploiting data-level parallelism. Data-level parallelism consists of performing the same operation to different data at the same time. Data must be of a uniform type and must need the same instruction behavior. SIMD basic unit is the vector, which consist of a row of individual numbers or scalars. Regular CPUs perform operations on scalars one at a time. However, SIMD instructions operate on all the scalars of a vector as a unit, performing the same operation on each scalar. For example, considering single-precision floating-point, which occupies 32-bit. Calculations in parallel can be done if data is grouped by 128-bit vector, allowing doing four single precision floating-point operations at the same time [36, 37]. So, the length of the individual vectors determines the number of elements of that type that can be worked with. Streaming SIMD Extensions (SSE) is an implementation of SIMD instructions that allows working with 128-bit vectors. Advanced vector extension (AVX), a more advanced implementation, allows working with 256-bit vectors (i.e., up to 8 floats can be processed in parallel). It is only available to processors which have Intel Sandy Bridge, AMD Bulldozer architecture or newer.

The use of SIMD instructions are disabled by default. Using the *gcc* compiler, we enable the generation of these instructions by adding the *-msseX* or *-mavx ags*, where X represents the SSE version number when compiling.

The code below shows the implementation of the emotional architecture using SIMD instructions:

```
#include <emmintrin.h>

#include <mmintrin.h>

...


int main(...) {

//one is a vector composed of 1.0 values,

// two is composed of 2.0 values,

// and zero is composed of 0.0 values

float *aux;

__m128 div, ptrPos, ptrNeg, ptr, ptr2, ptrEm, one, two, zero;

...


  posix_memalign( (void**)&fci, 16, n   * sizeof(float) );

  posix_memalign( (void**)&a,   16, n   * sizeof(float));

  posix_memalign( (void**)&m,   16, n/6 * sizeof(float));

...


//hyperbolic tangent
```

```
  for( i=0; i<n/4; ++i) {
    ptr    = _mm_load_ps( &a[4*i] );
    ptr    = _mm_mul_ps ( two, ptr );
    ptr    = fmath::exp_ps( ptr );
    ptrNeg = _mm_sub_ps ( ptr, one );
    ptrPos = _mm_add_ps ( ptr, one );
    div    = _mm_div_ps ( ptrNeg, ptrPos );
    _mm_store_ps( fci+4*i, div ); }

//reduction
  for( i=0; i<n/6; ++i ) {
// loading values
    ptr  = _mm_load_ps(&fci[4*i]);
    ptr2 = _mm_load_ps(&fci[4*(i+1)]);
    ptr2 = _mm_movelh_ps(ptr2, zero);
// horizontal add
    ptrEm = _mm_hadd_ps(ptr,ptr2);
    ptrEm = _mm_hadd_ps(ptrEm,zero);
    ptrEm = _mm_hadd_ps(ptrEm,zero);
    _mm_store_ps(&aux[0], ptrEm);
    m[i] = aux[0];
    j++;}
...}
```

Depending on which set of SIMD instructions are being used and its version, a different include directive must used in the code. In this case, the SSE instructions version 3.0 are used. To declare that a variable requires 128/256-bit registers, the m128/ m256 types should be used. The allocation of memory for data is done using the `posix_memalign((void**)&a, 16, n * sizeof(float))` function call, which ensures aligned data that leads to a better behavior. The `mm_load_ps(&app[4])` is one of the calls that are enabled by using the gcc -msse option flag. It will generate an assembler instruction that loads the first four members starting from the i^{th} pointer of a and stores them in ptr. Then, the computation of the hyperbolic tangent begins. As there is not an instruction to compute an exponential in SSE instructions, a call to a function of the `fmath` library was used [38]. In each iteration of the loop, the resulting data is stored with the `mm_store_ps` function. Finally, notice that the loop last n/4 iterations, due to each iteration performs 4 hyperbolic tangents in parallel.

During the reduction of the hyperbolic tangents in groups of six, the addition of the six elements is performed by using the “horizontal add" instruction. The `mm_hadd_ps(ptr, ptr2)` adds horizontally `ptr` and `ptr2` (i.e., it adds the values of its operands by pairs). The three horizontal adds allows performing the sum of up to 8 operands (six of them are used in our case). The result will be stored in the i^{th} position of `m`.

4.5. Evaluation

4.5.1. Robot application

The emotional processor is designed to tackle mobile robotic applications.

The multipurpose mobile robot performs activities such as diagnosis, transportation, cleaning, and surveillance, simultaneously. To define the emotional computational workload of the applications, a simulator of the robot environment is used (see Fig. 4.3). The simulator generates a large stock of scenarios to test the robotic platform while performing its activities.

As an example, Fig. 4.3 shows the result of an accident and the mobile robot trying to _x it. After the accident, there are multiple parts of a broken object and dust spots spread over the spatial area represented in the figure. To _x the accident (1) the robot (2) defines a set of sub-problems (3). It must pick the parts up and clean the spots.

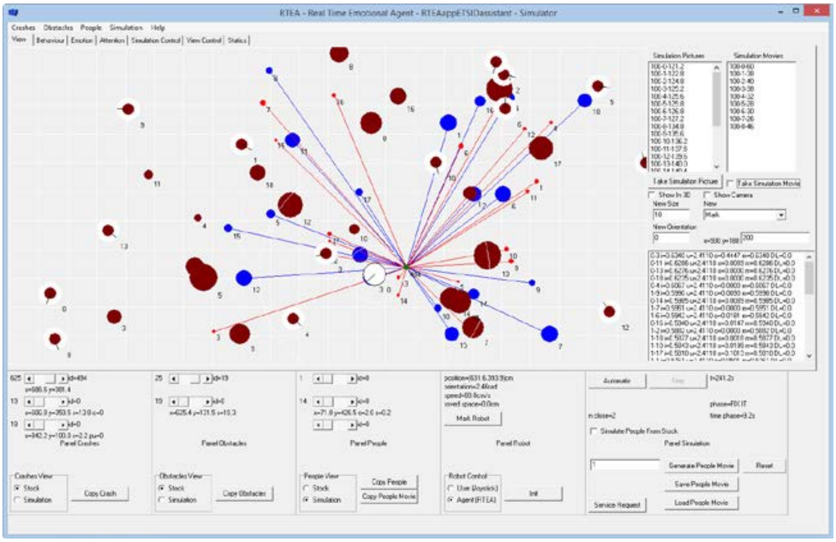

Fig. 4.3 Robot solving a crash in the operating area

The emotional system of the robot motivates every sub-problem considering several appraisals about the sub-problem situation: the importance, the probability of success, the urgency, and the opportunity.

The attention system of the robot uses these calculated motivation values to apply its attention policy. A set of people populates the accident scenario too (4). The simulator defines the behavior of these people as they move around the crash point. These people and other obstacles interfere the activities the robot performs. To guarantee the safety requirements and avoid collisions, the robot must perform the repairing activities adjusting its speed as the conditions of the environment change.

The robot speed, the number of objects in the operation area, and the collision risk factors define the attention cycle of the robot, Ta, which lies in the range of [0.1s , 0.5s].

Table 4.1 shows the robot speed values used in the experiments.

Table 4.1 Robot speed

Safe	Normal	Risky
0.1m/s	1m/s	2m/s

Table 4.2 Complexity of application problems

Simple	Normal	Complex
0.5Mopc	1Mopc	2Mopc

Table 4.2 shows the assumed values of the complexity for three types of applications, measured in required millions of emotion operations per attention cycle (Mopc). To obtain these values, applications of different complexities are run in a simulator environment and the number of emotions involved in each of the applications is calculated. A simple problem requires the execution of about 0.5Mopc, while a complex application involves the execution of 2Mopc.

The emotional state of the robot represents the ratio between the time spent to execute the emotional processes and the attention cycle (Ta). Three robot emotional states are considered in the experiments (i.e., relaxed, normal and stressed). In the relaxed mode, the robot dedicates less time to the emotional processing and more time to solve application problems, whereas in the stressed mode it is the contrary. In the relaxed mode, the time used to compute emotions is less than 10% of Ta. In the normal mode, it is assumed between 10% and 25%, and in the stressed mode it is between 25% and 40%, as shown in Table 4.3. A workload higher than 40% will not be acceptable because the applications processes are stalled and the robot cannot fulfill the objectives.

By combining the different problem complexities, robot speeds and emotional states, the computing power of the emotional architecture can be estimated, measured in MOPS (millions operations per second, see Section 4.1). Table 4.4 shows these requirements.

Table 4.3 Robot global emotional state

Relaxed	Normal	Stressed
< 0.1	(0.1, 0.25)	[0.25, 0.4]

Table 4.4 Required emotional processing power (MOPS)

Problem	Robot state	Robot speed 0.1m/s	1m/s	2m/s
Simple	Stressed	3	5	11
	Normal	4	8	17
	Relaxed	13	25	51
Normal	Stressed	6	11	19
	Normal	8	17	33
	Relaxed	26	49	99
Complex	Stressed	9	21	39
	Normal	17	33	68
	Relaxed	51	99	200

4.5.2. Evaluation framework

The parallel implementation of the emotional architecture proposed in this paper has been evaluated on different platforms. The version based on the use of multicore processors has been run on a Intel core i7 processor with 4 cores running at 2.93GHz (3.6GHz in turbo mode) [39]. For SIMD instructions, both versions using SSE and AVX instructions were evaluated. For GPUs, two experiments were run on two different graphic cards, an Nvidia GTX 9800 and an Nvidia GTX 670. Table 4.5 shows the characteristics of these GPUs. For comparison purposes, results for the sequential algorithm running on one core and for an implementation based on FPGAs (e.g., Stratix IV) are also shown.

The emotional based robot executing different applications, under the different environmental conditions, robot emotional state and dynamics, is evaluated.

Table 4.5 GPUs characteristics

Characteristic	GPU models	
	GTX 9800	GTX 670
Cuda cores	128	1344
Processor frequency (MHz)	675	980
Memory bandwidth (GB/s)	70.4	192.2

The evaluation is focused in the analysis of the performance, measured on MOPS.

4.5.3. Evaluation results

Fig. 4.4 to Fig. 4.12 show the results of evaluating the different implementation alternatives. In each figure, the bars represent the maximum computation capacity in MOPS that each processor, SIMD, GPU or FPGA can achieve, respectively. For each pair (complexity problem, robot emotional state), the robot speed imposes a minimum computational capacity required to solve a specific problem. This is shown as the horizontal lines in the figures. For instance, in the case of a simple problem and a relaxed robot (Fig. 4.6), if the speed is 2m/s, the minimum computation capacity required by the processors is 51 MOPS, while at 0.1m/s the required capacity is 13 MOPS (these bounds are the ones shown in Table 4.4). In general, for the same type of problems, at higher speeds, the computational requirements of the processors increase.

On the other hand, as the complexity of the problem increases, the processor computation requirements to solve the problem also increase. Moreover, for the same kind of problems, if the emotional robot state is becoming more stressed, then the computational requirements decrease because the time dedicated to the emotional computation is higher.

For a simple problem, and a stressed robot (Fig. 4.4), any implementation is able to fulfill the requirements in excess. This is also the case of a normal robot (Fig. 4.5).

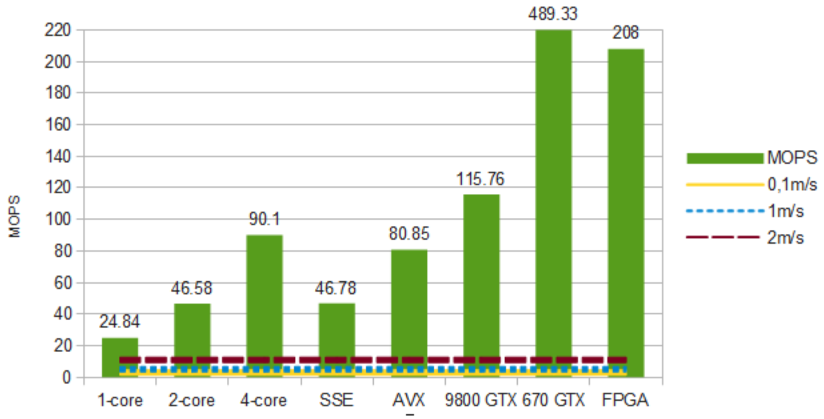

Fig. 4.4 Simple problem - stressed robot

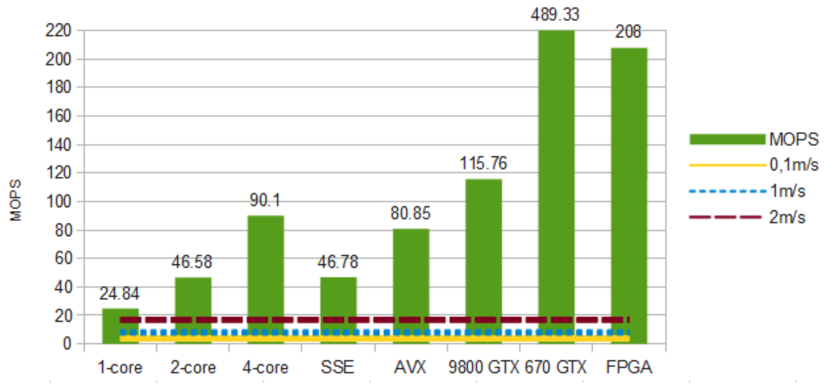

Fig. 4.5 Simple problem - normal robot

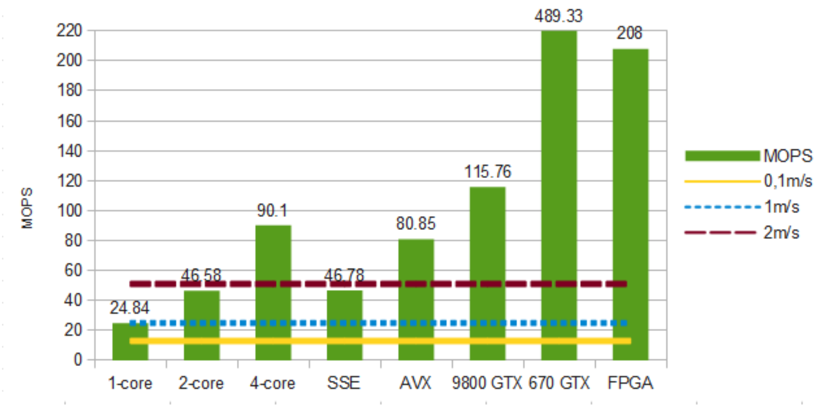

Fig. 4.6 Simple problem - relaxed robot

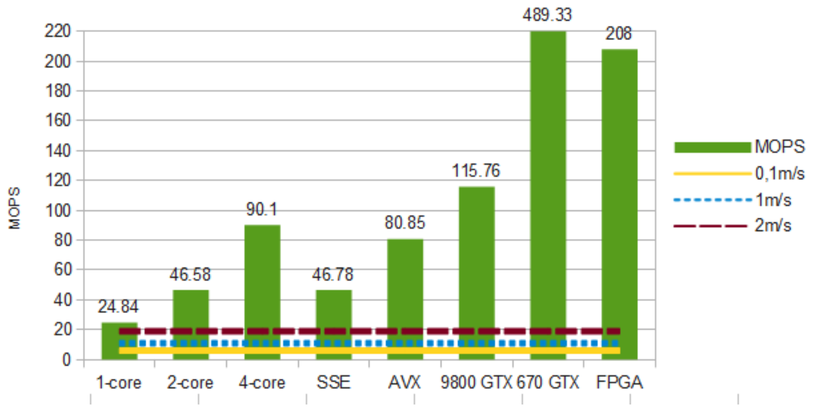

Fig. 4.7 Normal problem - stressed robot

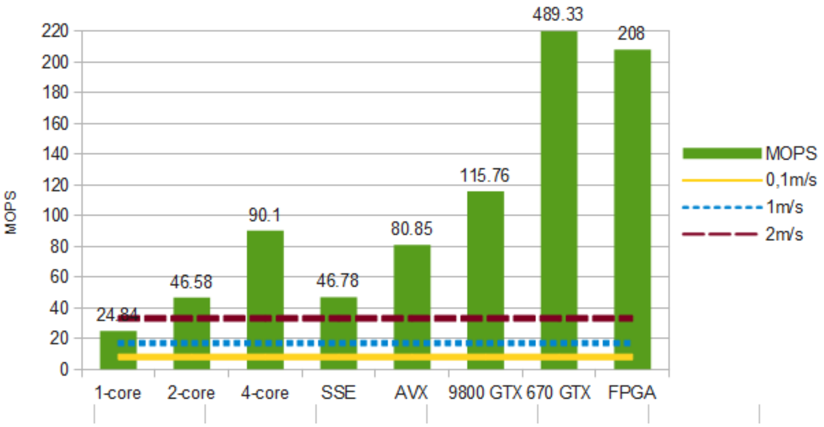

Fig. 4.8 Normal problem - normal robot

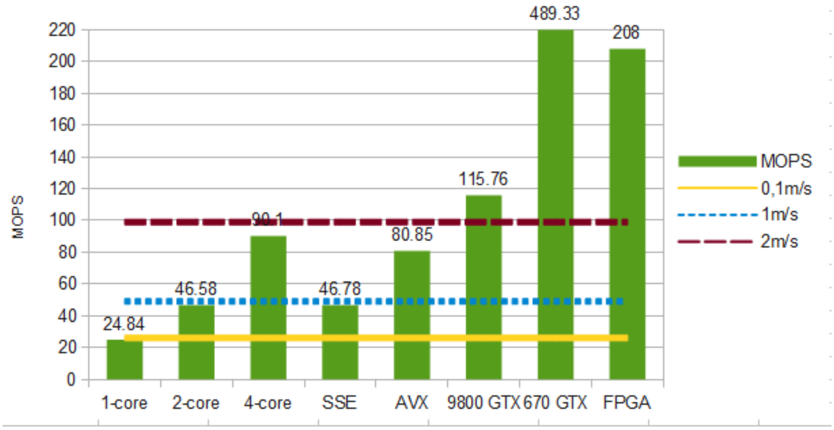

Fig. 4.9 Normal problem - relaxed robot

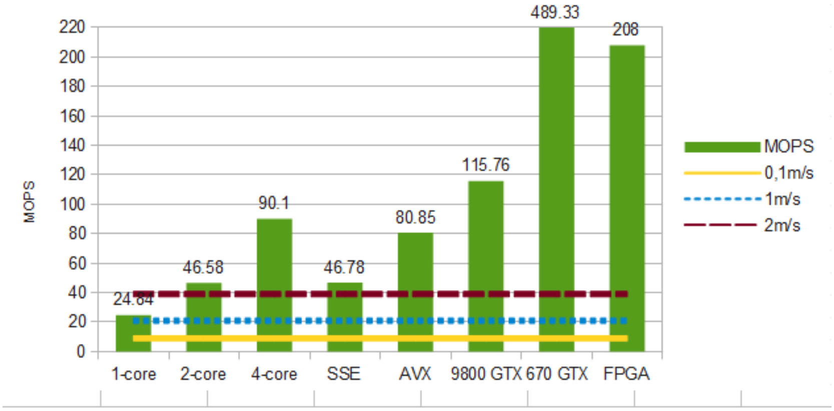

Fig. 4.10 Complex problem - stressed robot

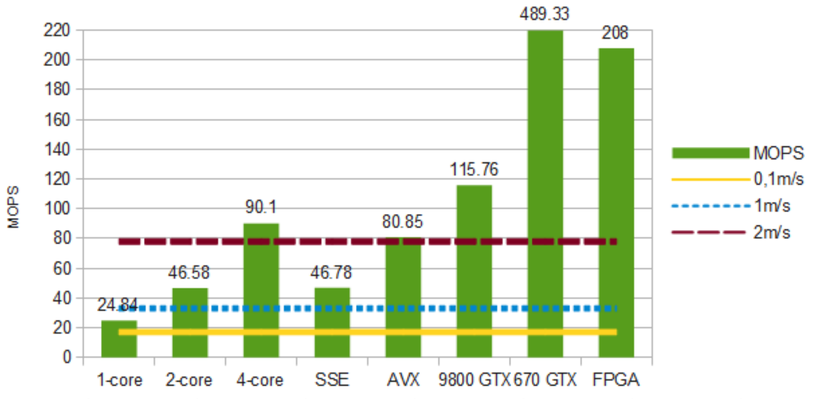

Fig. 4.11 Complex problem - normal robot

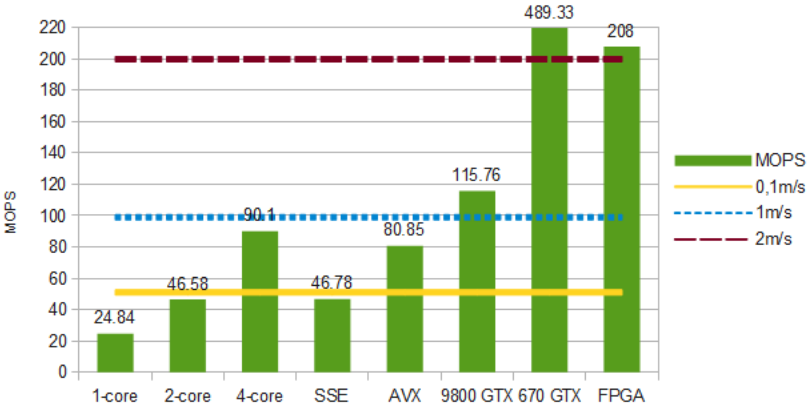

Fig. 4.12 Complex problem - relaxed robot

For a relaxed robot solving simple problems (Fig. 4.6) the sequential implementation of the emotional architecture only works for the low and medium speeds. For the highest speed, one-core implementation only achieves 24.84 of the 51 required MOPS.

The sequential implementation also works for a normal problem and a stressed robot (Fig. 4.7). However, it is unable to support a normal problem on a normal robot (Fig. 4.8) at high speed. Any of the parallel implementations are enough in this scenario. For a normal problem in a relaxed robot (Fig. 4.9), the results are quite different. Only GPU- and FPGA-based implementations can support the robot running at the highest speed. AVX-based and Quad-core implementations also support 1m/s speed. Two-core and SSE-based implementation supports only 0.1m/s.

For complex problems, the sequential implementation fulfills with a stressed robot running at 1m/s. Any parallel implementation is a good choice for a stressed robot (Fig. 4.10). For a normal robot (Fig. 4.11), any parallel implementation works up to 1 m/s. For the highest speed, only quad-core, AVX, GPU and FPGA based implementations work. For a relaxed robot (Fig. 4.12) moving at 2 m/s (the most constrained requirements) only the FPGA and one of the GPUs (GTX 670) are able to provide the required computational power.

As shown, only Nvidia GTX 670 and FPGA Stratix IV can tackle complex problems under the most constrained requirements: relaxed robot at maximum speed. However, in this case 670 GTX is a better election due to it outperforms Stratix IV in computational capabilities and its cost is much lower than Stratix IV. Nvidia GTX 9800 and Quad-core, which achieve approximately the same processing power, are the next more powerful solutions, hence they are a suitable election to solve less constrained problems than the previous one but still complex ones (e. g., complex problem and normal robot). SSE instructions are not able to tackle problems when the robot is relaxed and the robot maximum speed is required. Its performance is almost the same as a two-core processor. AVX instructions provide almost the same performance as a quad core processor and can tackle almost the same problems as the Nvidia GTX 9800. It must be noticed that SSE and AVX instructions allows increasing the computer computational capabilities at zero hardware cost.

4.6. Conclusions

Emotional architectures are being considered promising solutions to implement robots of the future. However these architectures have very high computational requirements, which consumes the computational power of the main robot controller. To reduce this consumption and allow the main controller solving more complex tasks, the parallelism of the emotional processes of the architecture have been exploited and their implementation on GPUs, multicore processors and using SIMD instructions have been tackled. A mobile robotic application -under different environmental, dynamic and emotional robot state conditions, implementing an emotional-based GPU architecture has been proposed.

The robotic application performances have been evaluated for Nvidia GTX 9800 and Nvidia 670, and the results are compared with a quad-Core processor (i.e., Intel i7 CPU 870 2.93GHz), SIMD instructions (i.e., SSE and AVX) and FPGA (Stratix IV). Results show that Nvidia GTX 670 and Stratix IV solve most complex problems under the most constrained requirements, but Stratix IV is much more expensive than GTX 670. Nvidia GTX 9800 and quad-core processors solve medium size problems, while AVX instructions obtains similar performance but without any additional hardware cost; however it requires a processor with Sandy Bridge architecture or newer. SSE instructions provides roughly the same performance as a dual-core and allows tackling some of the normal problems without any additional cost.

5. Publicación C:

Embedded Multicore Processors and SIMD Instructions for Emotional-based Mobile Robotic Agents

Francisco Almenar, Carlos Domínguez, Juan-Miguel Martínez, Houcine Hassan, Pedro López.

2014 IEEE International Conference on High Performance Computing and Communications.

DOI 10.1109/HPCC.2014.172

Abstract

Control architectures using emotions are useful for the implementation of robotic systems. The emotion processes regulates the execution of the robot behaviors to complete successfully its activities. In complex problems, the number of emotions explode, leading to situations where a single core is not capable of supporting the corresponding workload. In this paper, two solutions to this problem are presented: multicore processors and single instruction multiple data (SIMD) instructions, which are used to provide parallelism for the emotional processes. Different multicore processors and SIMD instruction sets are compared to analyze their suitability to cope with robotic applications. The applications are set-up taking into account different environmental conditions, robot dynamics and emotional states. Experimental results show that although SIMD instructions alone are not enough to undertake complex robotic problems, they allow obtaining some speed-up at zero cost, just by using processor intrinsic instructions. Dual-core processors show a

similar performance to SIMD instructions, while the use of quad-core processors allows solving medium and low size problems and most of the complex problems.

5.1. Introduction

Many research works [1], [2], [3], [4], [5] predict a growth in the number of intelligent robots in the industry and in our lives in the two next decades. They state that advanced robots capable of making decisions on their own as humans do, are still under development and the first prototypes will not start to appear until 2030. Some researches [2], [6] state that we are seeing the emergence of the first generation robots such as the demining robot Warrior manufactured by iRobot [7], which are able to solve simple tasks with little ability to adapt to the changing environment, and running their program code on a single-core processor. However, more intelligent features that robots could include such as decision-making are not yet developed in real robots.

Control architectures based on emotions are inspired on emotional natural agents. They are becoming promising solutions for the implementation of advanced robotic systems [3], [8], [9], [10], [28], because they facilitate the process of decision-making. They use the mechanism of emotion in organizing the behaviors, which has the following advantages: allow the robot to be autonomous to focus its attention on the most promising behavior, provide a bounded response time, which helps organizing the deliberative processes, sort the problems based on the expectations of success, autonomously adapt the computational load to the available processor capacity allowing solving problems of increasingly complexity, separate the decision from the action processes and use of subjective appraisal of the situation permit find always an alternative solution.

In this paper, an emotional robotic architecture for the control of complex mobile robot applications is used. In this model, two main types of processes coexist: behavior and emotion processes. The former solve the application problems (e.g., surveillance) while the latter use an emotional mechanism to motivate the robot behaviors.

Originally, all the processes of the emotional architecture, including the behaviors and the emotions were executed on a single-core computer (e.g., Intel@2,6GHz). The emotional processes must be applied to all problems/sub-problems of the robot agenda at every cycle of attention (e.g., 0.1s). As the agenda grows in high complexity level applications, the emotional workload increases significantly as well (e.g., 200 million operations per second (MOPS)). Each operation is a reduction function involving: an hyperbolic tangent function, a multiplication and a sum of up to 6 other functions. However, the control computer did not support this intensive workload because it could only execute up to 25MOPS.

Fortunately, there are some alternatives. Taking into account the inherent parallelism of the problem, this paper proposes the use of multi-core processors and SIMD instructions to implement the emotional system. All these alternatives do not need of any special hardware except from the ones that can be found in any modern computer.

The rest of the paper is organized as follows. Section 5.2 describes the problem to be solved and the sequential algorithm that implements it. Section 5.3 describes the alternative parallel implementations explored in this paper. Section 5.4 presents some evaluation results for different scenarios. Finally, some conclusions are drawn.

5.2. Emotional architecture

5.2.1. Emotional processes specification

An emotion is the process of appraising (observation) a situation and motivating a robot behavior to undertake this situation.

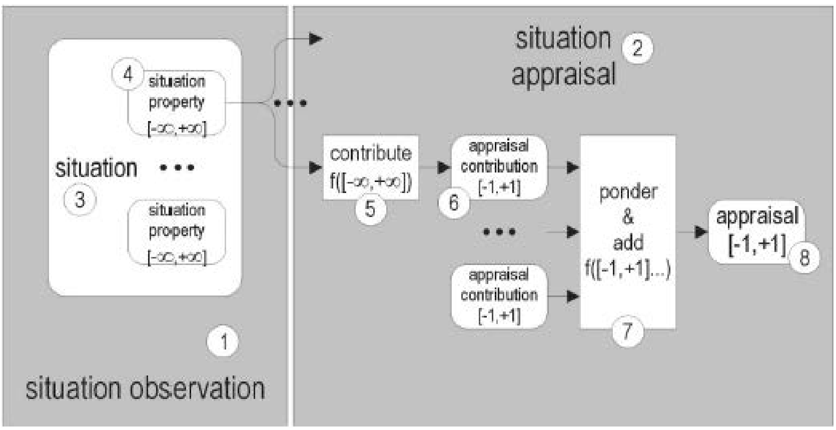

Fig. 5.1 Situation appraisal process

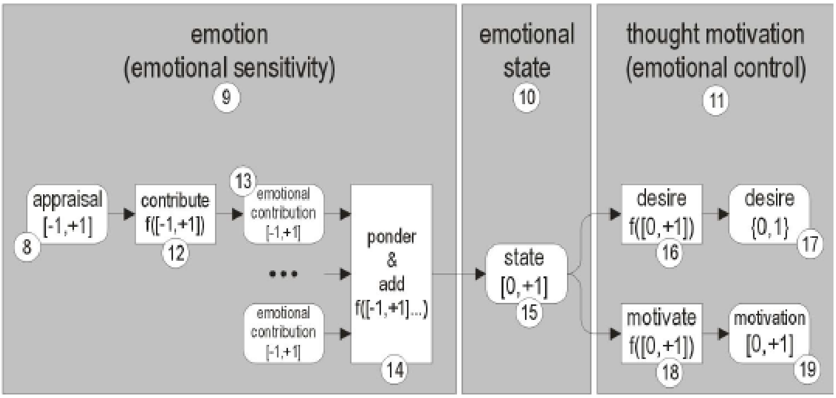

Fig. 5.2 Emotional motivation process

Fig. 5.1 and Fig. 5.2 detail the emotional motivation process. Fig. 5.1 shows the appraisal of the observed situation. Situations (3) are generated by observation processes (1) and are represented as real properties (4). The appraisal process (2) depends on the robot character. The character dynamically adjusts the parameters of this process. To calculate the appraisal of the situation (8) the robot weights and adds (7) a set of appraisal contributions (6), which are evaluated using contribution functions (5). Equation 5.1 represents the i^{th} situation appraisal, represented as follows:

$$a_i = \sum_{k=1}^{l} w_{ak} * f_{ak}(p_k) \tag{5.1}$$

Where: p_k is the k^{th} property of the situation, f_{ak} is the k^{th} contribution function, w_{ak} is the weight of the function and l is the number of appraisal contribution in range of 1 to 6. Fig. 5.2 shows the emotional motivation process. This process has two phases. First, an emotional activation (9) sets an emotional state (10), and second, an emotional response builds and motivates a behavior (11). The emotional contributions (13), evaluated with contribution functions (12), are weighted and added (14) to finally give an emotional state (15). The emotional contributions functions are defined in the real range [-1,+1] and the emotional state in the [0, +1] range. Every emotional state is labeled in the robot navigation problem (e.g., for “fear of collision”, a 0 level would mean “no fear”, while a 1 level would be “afraid”). The emotional response generates new desires (16, 17), and motivates the behavior to accomplish the desire (18, 19).

The j^{th} emotion is expressed as follows:

$$s_j = \sum_{i=1}^{l} w_{ci} * f_{ci}(a_i) \tag{5.2}$$

Where: s_j is the state of the j^{th} emotion, f_{ci} is the i^{th} contribution function, w_{ci} is the i^{th} is the weight of the function. The emotion contribution functions, f_{ci}, must have some properties such as a slight variation at the end of the range that tend to asymptotic values and abrupt variations around an inflection point in the center of the range. These properties are found in the hyperbolic tangent functions, which are used to represent the contribution functions:

$$f_{ci}(x) = th(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{5.3}$$

Where x is the appraisal value a_i when calculating the emotion. To allow adjusting the hyperbolic function, Equation 5.3 is transformed in the following function:

$$th^*(x) = \left(\frac{e^{2(x - x_0)\delta_y} - 1}{e^{2(x - x_0)\delta_y} + 1} \right) * \delta_y \tag{5.4}$$

Where the parameters x_0, y_0, δ_x and δ_y permit to translate and scale the contribution function. These emotions are grouped in the emotional system and have the structure shown in Fig. 5.3. The emotional system gets, at a given instant, inputs from a set of n situation appraisals (e.g., 2M, M refers to millions) and produces a set of K motivations (e.g., 0.5M). The hyperbolic tangent is applied to each appraisal and its result is multiplied by a weight. Each emotion can be composed of up to 6 different contribution functions. The obtained emotion can pass through a final function (f_i). In this paper, though, the identity function is used (i.e. no post-processing is done). The total number of these situation appraisals in the emotional system depends on the complexity of the problem, the environment conditions and the robot dynamics.

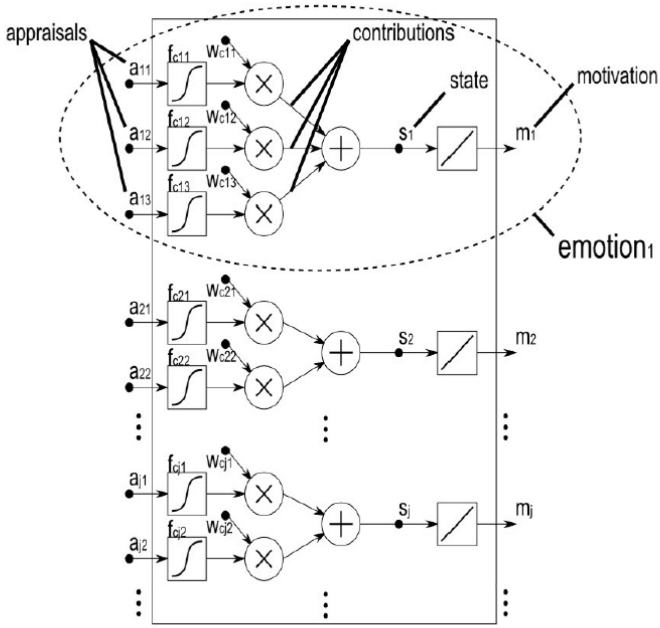

Fig. 5.3 Emotional system structure

In the experiments of the multi-objective robotic applications, this number reaches a value of about 200MOPS. These operations can be computed in parallel since they are independent. However, in the initial implementation they were executed sequentially in a host computer controlling the robot. Due to their highly computational requirements, the capacity of the host processor was exceeded, being unable to fulfill the robot objectives.

In the next section, this paper exploits the inherent parallelism of the problem, proposing a parallel implementation that takes advantage of the hardware already available in modern computers to implement the emotional system.

5.2.2. Sequential Algorithm

In this paper, the hyperbolic tangent is used to implement the emotional system of the robot (as shown in Section 5.2.1). The following fragment of code shows an implementation in the C programming language. For the sake of simplicity, the translation and scaling factors and the weight of each contribution are not shown. This sequential code will be the base for the parallel code that will be explained later.

```
for(i=0;i<n;i++) {
   fci[i] = (exp(2*a[i])-1)/(exp(2*a[i])+1); }
for (i=0;i<n/6;i++) {
   for(j=0; j<6; j++) {
      acum += fci[6*i+j]; }
   m[i]= acum; }
```

In the following section, different approaches to exploit the exhibited parallelism will be shown.

5.3. Parallel implementation of the emotional architecture

5.3.1. Multicore processors

The first parallel implementation of the emotional architecture is carried out using multicore processors. Multicore processors with several number of cores are standard in today's computers. The availability of several cores allows to execute in parallel several threads. To generate the parallel threads, OpenMP (OMP) is used. OMP is an API that supports multi-platform shared memory multiprocessing programming in C, C++, and Fortran. This API modifies the run-time behavior to obtain thread-level parallelism. OMP relies on directives written by the programmer that communicate the compiler what can be executed in parallel. Parallelism is obtained by forking a master thread into a specified number of slave threads, these slave threads receive a part of the task that the master thread has to perform, allowing threads to run concurrently. To indicate that a loop can be executed in parallel, the preprocessor directive #pragma omp parallel for is written just before the for loop. In this case, the counter of the inner loop (j) and the auxiliary reduction variable (acum) are declared as private, due to there are not shared variables. The code below shows the OMP parallel version of the emotional system:

```
#include <omp.h>

...

#pragma omp parallel for private(j,acum)
for( i=0; i<n/6; i++) {
  acum = 0;

  for( j=0; j<6; j++) {
    fci[6*i+j] = (exp( 2*a[6*i+j])-1 ) / (exp( 2*a[6*i+j])+1 );
    acum += fci[6*i+j]; }

  m[i] = acum; }

...
```

5.3.2. SIMD instructions

Secondly, we propose the use of the SIMD (Single-Instruction Multiple-Data) instructions that are part of the ISA processors since the MMX instruction set extensions were introduced by INTEL in 1999. For a better comparison, we have implemented and evaluated a SIMD algorithm version that runs in only one core, even though it is possible to combine SIMD and multicore parallelism capabilities to obtain better results. SIMD instructions allow exploiting data-level parallelism. Data-level parallelism consists on performing the same operation to different data at the same time. Data must be of an uniform type; this data must have the same instruction behavior. SIMD basic unit is the vector, which consist of a row of individual numbers or scalars. Regular CPUs perform operations on scalars one at a time. However, SIMD instructions operate on all the scalars of a vector, performing the same operation on each scalar. For example, considering single-precision floating-point, which represents 32-bit. Calculations in parallel can be done if data are grouped by 128-bit vector, allowing performing four single-precision floating-point operations at the same time [36], [37]. So, the length of the individual vectors determines the number of elements of that type that can be processed. Streaming SIMD Extensions (SSE) is an implementation of SIMD instructions that allows operating with 128-bits vectors. Advanced vector extension (AVX), a more advanced implementation, allows working with 256-bits vectors (i.e., up to 8 floats can be processed in parallel). It is only available to processors which have Intel Sandy Bridge, AMD Bulldozer architecture or newer.

The use of SIMD instructions are disabled by default. Using the *gcc* compiler we can enable this instructions by adding the *-msseX* or *-mavx* flags, where *X* represents the SSE version number.

The code below shows the implementation of the emotional architecture using SIMD instructions:

```
#include <emmintrin.h>
#include <mmintrin.h>
...

int main(...) {
//one is a vector composed of 1.0 values,
// two is composed of 2.0 values,
// and zero is composed of 0.0 values
float *aux;
__m128 div, ptrPos, ptrNeg, ptr, ptr2, ptrEm, one, two, zero;

...

  posix_memalign( (void**)&fci, 16, n   * sizeof(float) );
  posix_memalign( (void**)&a,   16, n   * sizeof(float));
  posix_memalign( (void**)&m,   16, n/6 * sizeof(float));
...

//hyperbolic tangent
  for( i=0; i<n/4; ++i) {
    ptr    = _mm_load_ps( &a[4*i] );
    ptr    = _mm_mul_ps ( two, ptr );
    ptr    = fmath::exp_ps( ptr );
    ptrNeg = _mm_sub_ps ( ptr, one );
    ptrPos = _mm_add_ps ( ptr, one );
```

```
    div    = _mm_div_ps ( ptrNeg, ptrPos );
    _mm_store_ps( fci+4*i, div ); }

//reduction
  for( i=0; i<n/6; ++i ) {
// loading values
    ptr  = _mm_load_ps(&fci[4*i]);
    ptr2 = _mm_load_ps(&fci[4*(i+1)]);
    ptr2 = _mm_movelh_ps(ptr2, zero);
// horizontal add
    ptrEm = _mm_hadd_ps(ptr,ptr2);
    ptrEm = _mm_hadd_ps(ptrEm,zero);
    ptrEm = _mm_hadd_ps(ptrEm,zero);
    _mm_store_ps(&aux[0], ptrEm);
    m[i] = aux[0];
    j++; }
... }
```

Depending on which set of SIMD instructions is being used and its version, a different *include* directive must be used in the code. In this case, the SSE instructions version 3.0 is applied. To declare that a variable requires 128/256-bit registers, the m128/ m256 types should be used.

The allocation of memory for data is solved using posix_memalign((void**)&a, 16, n * sizeof(float)), which ensures aligned data that leads to a better behavior. mm_load_ps(&app[4]) is one of the calls that are enabled by using the gcc *−msse* flag option. It will generate an assembler instruction that loads the first four members starting from the i^{th} pointer of a and stores them in ptr. Then, the computation of the hyperbolic tangent begins. As there is not an instruction to calculate an exponential function in SSE instructions, a call to a function of the *fmath* library has been done [38]. In each iteration of the loop, the resulting data is stored with the mm_store_ps function. Finally, notice that the loop last n/4 iterations because each iteration performs 4 hyperbolic tangents in parallel.

During the reduction of the hyperbolic tangents in groups of six, the addition of the six elements is performed by three horizontal adds. The mm_had_ps(ptr, ptr2) adds horizontally ptr and ptr2 (i.e., it adds the values of its operands by pairs). The three horizontal adds allows performing the sum of up to 8 operands (six of them are used in our case). The result will be stored in the i^{th} position of m.

5.4. Evaluation

5.4.1. Robot application

The emotional processor is designed to tackle mobile robotic applications. The multipurpose mobile robot performs activities such as diagnosis, transportation, cleaning, and surveillance, simultaneously.

To define the emotional computational workload of the applications, a simulator of the robot environment was used. This simulator allows defining different scenarios where the robot can tackle the aforementioned activities. The robot speed, the number of objects in the operation area, and the collision risk factors define the attention cycle of the robot, Ta, which lies in the range of [0.1s , 0.5s]. Table 5.1 shows the values of the robot speed.

Table 5.1 Robot speed

Safe	Normal
0.1m/s	1m/s

Table 5.2 shows the assumed values of the complexity for three types of applications, measured in required millions of emotion operations per attention cycle (Mopc). To obtain these values, applications of different complexities are run in a simulator environment and the number of emotions involved in each of the applications is calculated.

A simple problem requires the execution of about 0.5Mopc, while a complex application involves the execution of 2Mopc.

Table 5.2 Complexity of application problems

Simple	Normal	Complex
0.5Mopc	1Mopc	2Mopc

The emotional state of the robot represents the ratio between the time spent to execute the emotional processes and the attention cycle (Ta). Three robot emotional states are considered in the experiments (i.e., relaxed, normal and stressed). In the relaxed mode, the robot dedicates less time to the emotional processing and more time to solve application problems, whereas in the stressed mode it is the contrary. In the relaxed mode, the time used to compute emotions is less than 10% of Ta. In the normal mode, it is assumed between 10% and 25%, and in the stressed mode it is between 25% and 40%, as shown in Table 5.3. A workload higher than 40% will not be acceptable because the applications processes are stalled and the robot cannot fulfill the objectives.

Table 5.3 Robot global emotional state

Relaxed	Normal	Stressed
< 0.1	(0.1, 0.25)	[0.25, 0.4]

By combining the different problem complexities, robot speeds and emotional states, the computing power required in the emotional architecture can be estimated, measured in MOPS (see section 5.2.1). Table 5.4 shows these requirements.

Table 5.4 Required emotional processing power (MOPS)

		Robot speed	
Problem	Robot state	0.1m/s	1m/s
Simple	Stressed	3	5
	Normal	4	8
	Relaxed	13	25
Normal	Stressed	6	11
	Normal	8	17
	Relaxed	26	49
Complex	Stressed	9	21
	Normal	17	33
	Relaxed	51	99

5.4.2. Evaluation framework

The parallel implementation of the emotional architecture proposed in this paper has been evaluated on different platforms. The version based on the use of multicore processors has been run on a Intel core i7 processor with 4 cores running at 2.93GHz (3.6GHz in turbo mode). For SIMD instructions, both versions using SSE and AVX instructions were evaluated. For comparison purposes, results for the algorithm running on one and two cores are also shown.

The emotional based robot executing different applications, under the different environmental conditions, robot emotional state and dynamics, is evaluated.

The evaluation is focused in the analysis of the performance, measured on MOPS (contributions operations per second, i.e., millions of hyperbolic tangents per second, see Section 5.1).

5.4.3. Evaluation results

Fig. 5.4 to Fig. 5.12 show the results of evaluating the different implementation alternatives, considering 3 complexity problems (Simple, Normal and Complex) and 3 robot emotional state levels (Stressed, Normal and Relaxed), for 2 different values of robot speeds (0.1m/s, 1m/s). In each figure, the bars represent the maximum computation capacity in MOPS that each processor or SIMD can achieve, respectively. For each pair (complexity problem, robot emotional state), the robot speed imposes a minimum computational capacity required to solve a specific problem. This is shown as the horizontal lines in the figures. For instance, in the case of a simple problem and a relaxed robot (see Fig. 5.6), if the speed is 1m/s, the minimum computation capacity required by the processors is 25MOPS, while at 0.1m/s the required capacity is 13MOPS (this bounds are the ones shown in Table 5.4). In general, for the same type of problems, at higher speeds, the computational requirements of the processors increase. On the other hand, as the complexity of the problem increases, the processor computation requirements to solve the problem also increase.

Moreover, for the same kind of problems, if the emotional robot state is becoming more stressed, then the computational requirements decrease because the time dedicated to the emotional computation is higher, and hence less computing power is claimed to the co-processor devices.

Quad-core processors are the more powerful solutions, hence they are a suitable election to solve less constrained problems still cannot solve the most constrained one (e.g., complex problem and relaxed robot). SSE instructions are not able to tackle problems when the robot is relaxed and the problems are normal or complex. SSE performance is almost the same as a Two-core processor. AVX instructions provide almost the same performance as a Quad-core processor. It must be noticed that SSE and AVX instructions allows increasing the computer computational capabilities at zero hardware cost.

Fig. 5.4 Simple problem - stressed robot

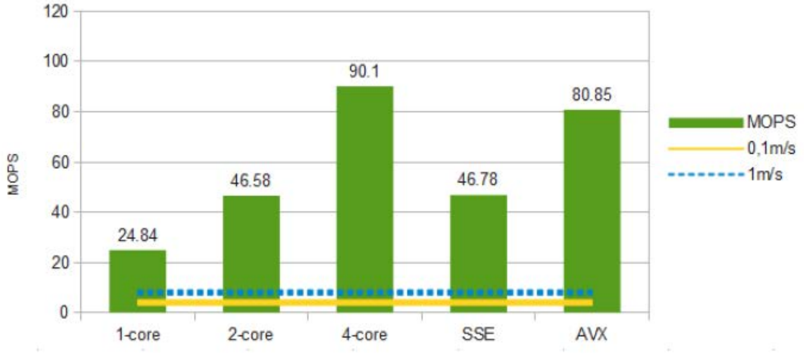

Fig. 5.5 Simple problem - normal robot

Fig. 5.6 Simple problem - relaxed robot

Fig. 5.7 Normal problem - stressed robot

Fig. 5.8 Normal problem - normal robot

Fig. 5.9 Normal problem - relaxed robot

Fig. 5.10 Complex problem - stressed robot

Fig. 5.11 Complex problem - normal robot

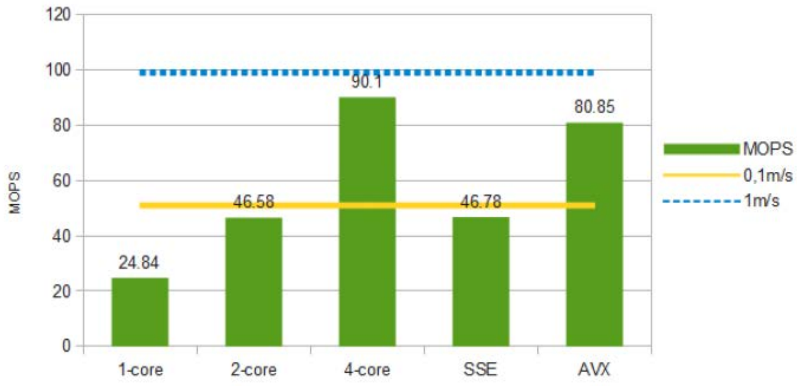

Fig. 5.12 Complex problem - relaxed robot

5.5. Conclusions

Emotional architectures are being considered promising solutions to implement robots of the future, however these architectures have very high computational requirements, which consumes the computational power of the main robot controller.

To reduce this consumption and allow the main controller solving more complex tasks, the parallelism of the emotional processes of the architecture have been exploited and their implementation on multicore processors and SIMD instructions have been tackled. A mobile robotic application - under different environmental, dynamic and emotional robot state conditions is evaluated. Quad-core processors solve all the problems except the most constrained one, while AVX instructions obtains similar performance but without any additional hardware cost. However, a processor with Sandy Bridge architecture or newer is required in order to be able to use this parallel characteristics. SSE instructions provides roughly the same performance as a dual-core and allows tackling some of the normal problems without any additional cost.

Acknowledgement

This work was supported in part by the Spanish Ministerio de Economía y Competitividad (MINECO) and by FEDER funds under Grant TIN2012-38341-C04-01 and by PAID/2012/325 Vicerrectorado de Investigación, Innovación y Transferencia, Universitat Politècnica de València.

6. Publicación D:

Multicore Parallel Implementation of Agent Emotional Processes

Carlos Domínguez, Houcine Hassan, Vicent Mayans, Alfons Crespo.

2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications.

DOI 10.1109/TrustCom.2013.248

Abstract

Emotional agents are becoming promising technologies for real-time control applications (i.e., robotics). The principal controllers of the agent are the emotional processes that decide the selection of behaviors to fulfill the objectives. The number of emotional processes increases with the complexity of the application, limiting the processing capacity of a mono-core processor to solve complex problems. A costly solution would be the use of HPC computers to solve complex problems. However, the possibilities of parallelization of the emotional processes permit their execution in parallel on multicore processors, enabling the agent to solve problems of higher complexity at a low cost. This paper presents the implementation of the emotional processes of a robotic agent in a multicore processor.

To this end, the parallel emotional processes are identified and characterized, and a real-time system based on the EDF scheduling policy, to execute the emotional workload on the multicore, is proposed. In the experiments, mobile robotic applications are set-up taking into account different environmental conditions, robot dynamics and emotional states. The applications are run on multicore multithreaded processors depending on the workload requirements of

the applications. Results show that only a three core with 6 threads can tackle complex problems in the worst emotional conditions, a dual-core solves less constrained problems under normal emotional conditions and a single-core processor is only suitable for simple problems under the best conditions.

6.1. Introduction

Control architectures based on emotions are becoming promising solutions for the implementation of advanced robotic systems [3, 8, 9, 10, 28]. Emotional control architectures have been inspired on emotional natural agents, and use emotional based reactive processes to motivate the behavior of the robot. In these models, operational processes of emotional control system consist of: (i) establishing emotional levels from subjective appraisals of situations, (ii) establishing levels of motivation of the problem-solving behavior (application processes) from the emotional levels, and (iii) the attention of the application process from motivation levels. These operational processes must be applied to all application problems/subproblems on the agenda of the robot every cycle of attention, besides of solving the processes of the application. As the agenda can grow significantly in real applications with medium-high levels of complexity (e.g., a multi-purpose mobile robot performing transportation, diagnosis, cleaning, surveillance simultaneously), the workload of the operational processes increases as well. This situation can lead to limit the capacity of a mono-processor to solve only simple problems of the robot. A typical but costly solution to this problem is to use HPC computers to solve more complex robotic applications. Taking into account the potential possibilities of parallelization of the emotional processes, and the advent of the multicore era, a low cost solution is proposed in this paper which consists of using multicores multithreaded processors [40] (1, 2 and 3 cores and up to six hardware threads) depending on the workload requirements of the robotic application. The multicore solution will tackle (i) more dynamic problems where the attention cycle is shortened and/or (ii) improve the productivity of the simultaneously number of problems to solve.

In this paper, multicore processors are used to implement the emotional processes. To this end, the potential parallel emotional processes are identified and characterized, and a real-time system based on the EDF (Earliest Deadline First) scheduling policy [41], to execute the emotional workload on the multicore, is proposed. The different processes of the agent (belief, behavior, attention and emotion), in a first step are fairly distributed among the cores using the WF policy [44]. One of the cores is dedicated to the application processes and the remaining 3 cores implement the emotional system. In a second step, within the scope of each core, a local EDF algorithm schedules the processes belonging to that core.

The cores share a memory called the blackboard [45]. Through this memory the processes read and write the important data as the appraisals, emotional states and motivations. Also it allows, using shared variables, to synchronize the different processes.

In the experiments, different application problems of varying complexity levels (simple, normal, complex), under distinct environmental conditions and different robot emotional states (relaxed, normal, stressed), are implemented. The applications are run on multicore multithreaded processors (1, 2 and 3 cores with up to two hardware threads per core) depending on the workload requirements of the applications. Results show that only a three-core with 6 threads can tackle complex problems in the worst emotional conditions, a dual-core with 4 threads solves less constrained problems under normal emotional conditions and a single-core processor is only suitable for simple problems under the best emotional conditions.

The rest of the paper is organized as follows: section 6.2 reviews the state of the art on emotional architectures; section 6.3 describes the general characteristics of the emotional control architecture; section 6.4 describes the robotic application and the multicore-based implementation; experimental results are discussed in section 6.5; finally, conclusions are sum-up in section 6.6.

6.2. Related work

Different control architectures based on emotions are found in the bibliography. Arkin et al. [17] develop algorithms based on the emotion of deception to control robots. The authors are inspired on the behavior of deception observed in animals or humans. In the simulations they show that robots including this emotion are more effective than traditional ones. Salichs [18] proposes a decision-making system based on drives and motivations, but also emotions and self-learning. The agent's goal is to learn to behave through interaction with the environment, using reinforcement learning, and maximizing their welfare. Lee-Johnson et al. [8] develop a hybrid architecture reactive/deliberative that incorporates artificial emotions to improve decision-making and actions of a mobile robot. Emotions are active at different levels of the architecture and serve to modulate the decisions and actions of the robot. Damiano [2] presents a model where decision-making is based on a motivational system. The motivations are dependent on the value of the need that have to be satisfied and a stimulus incentive. Once they calculate all the values, the highest motivation activates and organizes the behavior so as to satisfy the most urgent need. The aforementioned emotional models are implemented on high performance computer with a corresponding rise of the cost, unlike the proposal of this paper, which consists of the use of low cost multicore processors to parallelize the execution of the emotional processes to reduce the cost of implementation of emotional based agent architectures.

6.3. Emotional control architecture

In this section, the complexity of the emotional control architecture is described and the potential parallelization of the emotional processes workload is discussed.

6.3.1. Real-time emotional control architecture overview

A behavior of the emotional control architecture is based on a problem resolution process [42]. A problem arises when the robot desires a new situation different from the current one. Every new desire starts an associated thought, which defines a context for the execution of observation, decision and action processes related to the problem to be solved. These problem-domain processes are known as application processes. The operational processes, using an emotional mechanism, motivate the application processes, so that they can get the agent attention and be executed. Fig. 6.1 shows the information flow in the emotional control architecture [43].

Ellipses represent concepts and squares represent processes. The bold labeled arrows represent the main paths. The path labeled as (a) connects sensors and motors devices, from the perception to the action. The processes in this way are the application ones, and they flow through two subways, the deliberative-way (b) and the reactive-way (c). Reactive processes have a guaranteed response time, which is usually short. The response time of the deliberative processes however is usually longer and cannot be guaranteed.

The other paths are for the operational emotional processes, which generate and motivate new desire-thought pairs. These are: arouse-desire-procure (d) and motivate-attention (e). Every thought is motivated by an emotion, and the motivation that it gets, plays later an important role in the attention policy of the controller.

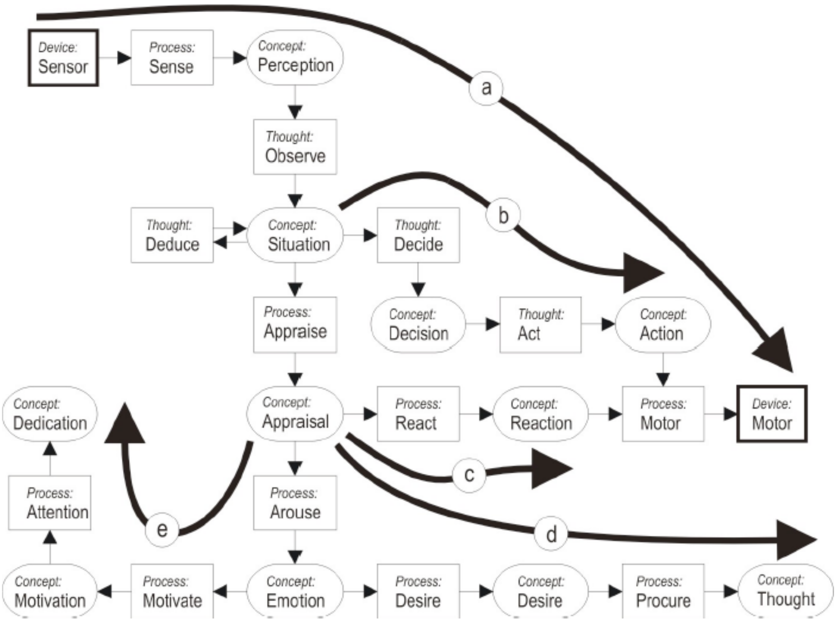

Fig. 6.1 Information flow in the emotional control architecture

6.3.2. The emotional processes

An emotion is the process of appraising (observation) a situation and motivating a behavior of the agent to afford this situation (see Fig. 6.2).

Fig. 6.2 Emotional process

Fig. 6.3 and Fig. 6.4 detail the emotional motivation process. Fig. 6.3 shows the subjective appraisal of the observed situation.

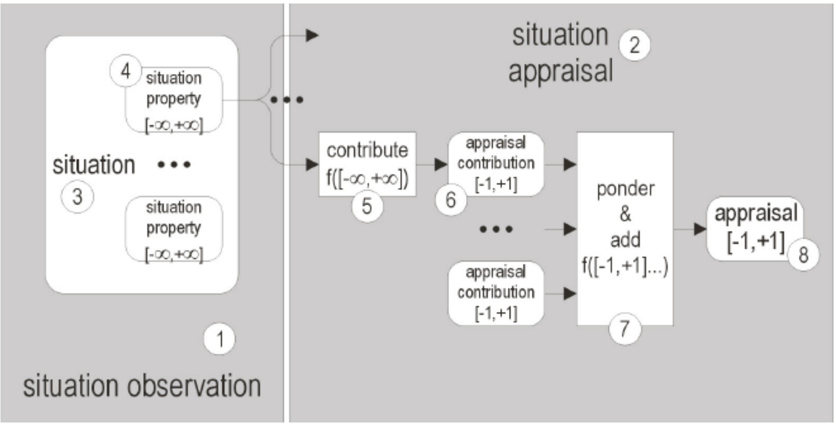

Fig. 6.3 Situation appraisal process

Situations (3) are generated by observation processes (1) and are represented as sets of interrelated real properties (4). The appraisal process (2) depends on the robot character. The character dynamically adjusts the parameters of this process. To calculate the appraisal of the situation (8) the agent ponders and adds (7) a set of appraisal contributions (6), which are evaluated using contribution functions (5). Fig. 6.4 shows the emotional motivation process. This process has two phases: first, an emotional activation (9) sets an emotional state (10), and second, an emotional response builds and motivates a desire-thought pair (11). The intermediary emotional states, being explicitly defined, can be used as key parameters to adjust the agent character, tuning this way its emotional sensibility and self-control. The structure of the emotional activation process (9) is similar to the situation appraisal process. The emotional contributions (13), evaluated with contribution functions (12), are pondered and added (14) to finally give an emotional state (15). The emotional contributions are defined in the real range [-1,+1] and the emotional state in [0, +1]. Every emotional state is labeled in the robot navigation problem (e.g., “fear of collision”). Then, a 0 level would mean “no fear at all”, while a 1 level would be “absolutely afraid”. The emotional response (11) generates new desires (16) and (17), and motivates these latters (18) and (19).

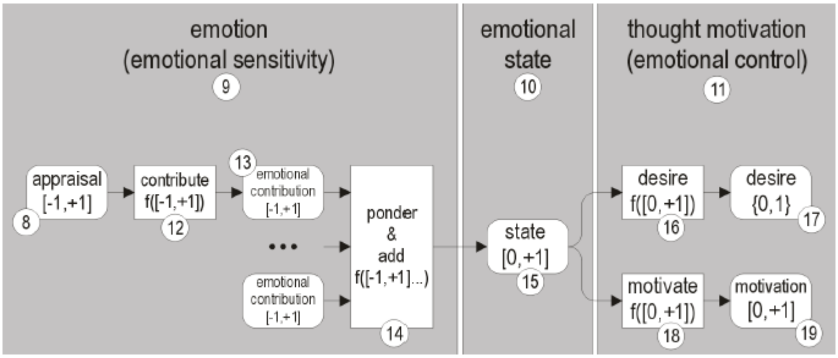

Fig. 6.4 Emotional motivation process

The emotional process can be represented by Equation 6.1:

$$s = \sum_i w_i \cdot f(v_i) \tag{6.1}$$

Where:

- s is the emotional state
- v_i is the i^{th} appraisal of the situation
- f is the emotional contribution function
- w_i is the weight of the contribution $f(v_i)$

In the "fear of collision" example, the E_{fear} emotion would set the motivation of the obstacle avoidance behavior. The robot would appraise the variables *distance* and *speed*, which define the current situation, using the appraisal functions f_{ad} y f_{as} (see Equation 6.2)

$$\left.\begin{array}{r} distance\_app = f_{ad}(distance) \\ speed\_app = f_{as}(speed) \end{array}\right\} \tag{6.2}$$

The emotional state s_{fear} would be evaluated as shown in Equation 6.3, where f_{cd} and f_{cs} are the contribution functions.

$$s_{fear} = w_d \cdot f_{cd}(distance\_app) + w_s \cdot f_{cs}(speed\_app) \qquad (6.3)$$

6.3.3. Emotional computational workload

The industrial robot has an agenda with the set of pending problems (e.g. transport, surveillance). Each problem is decomposed into many sub-problems which in turns has to execute the above described emotional motivation processes.

These emotional processes must be executed no matter if the sub-problem is being dispatched or is pending. This process is repeated in each attention cycle T_a (see Fig. 6.5). The attention cycle is the observation-decision-action cycle of the robot. In Fig. 6.5, t_e is the processing time of the emotional processes. The parameter that relates the emotional processing time and the attention cycle is $\varepsilon = \frac{t_e}{T_a}$. Ɛ represents the robot global emotional state.

Fig. 6.5 Attention cycle

The computational workload of the emotional processor is defined as the number of emotional contributions to be evaluated every attention cycle. Since the robot agenda can grow in applications with medium to high level complexity, (e.g. industrial mobile robot performing transportation, cleaning and surveillance simultaneously), the computational workload of the emotional processor can grow substantially. These high computational requirements of the emotional system can surpass the capacity of a mono-processor leading to situations where the robot is unable to execute the application processes and solve the problems.

Moreover, these emotional processes can be computed in parallel since they are independent for a given attention cycle, in the sense that they perceive the situation and have to evaluate it and generate an emotional state in the same attention cycle. Therefore, multicore processors are suitable to execute these operations and increase the resolution capacity of the agent. To this end, two actions are applied: (i) reducing T_a and maintaining Ɛ, in this case more time critical dynamic problems where the attention cycle is shorter can be solved (e.g., increase robot speed). (ii) reducing Ɛ and maintaining T_a, in this case the productivity of the number of problems to solve simultaneously, is improved.

6.4. Emotional processes design

6.4.1. Industrial robot application

The multicore emotional processor is designed to tackle mobile industrial robot applications. The robot performs activities such as: transport of objects, search of pieces, surfaces maintenance and facilities surveillance (see Fig 6.6).

Fig. 6.6 Robot in the operating area

The emotional processes are implemented on the multicore processor in order to increase the capacity of the robotic agent to solve more complex problems.

To establish the emotional computational workload, a simulator of the problem environment has been used (see Fig. 6.7). The robot speed, the number of objects in the operation area, and the collision risk factors have been considered to set the attention cycle of the robot, T_a , in the range [0.1s, 0.5s]. Table 6.1 shows the complexity of the tackled problems (measured in millions of emotional operations per attention cycle, Mopc) and Table 6.2 shows three possible emotional states that the robot faces when affording the resolution of the different problems.

Table 6.1 Complexity of the problem

Simple	Normal	Complex
0.5Mopc	1Mopc	2Mopc

Table 6.2 Robot global emotional state Ɛ

Relaxed	Normal	Stressed
< 0.1	(0.1, 0.25)	[0.25, 0.4]

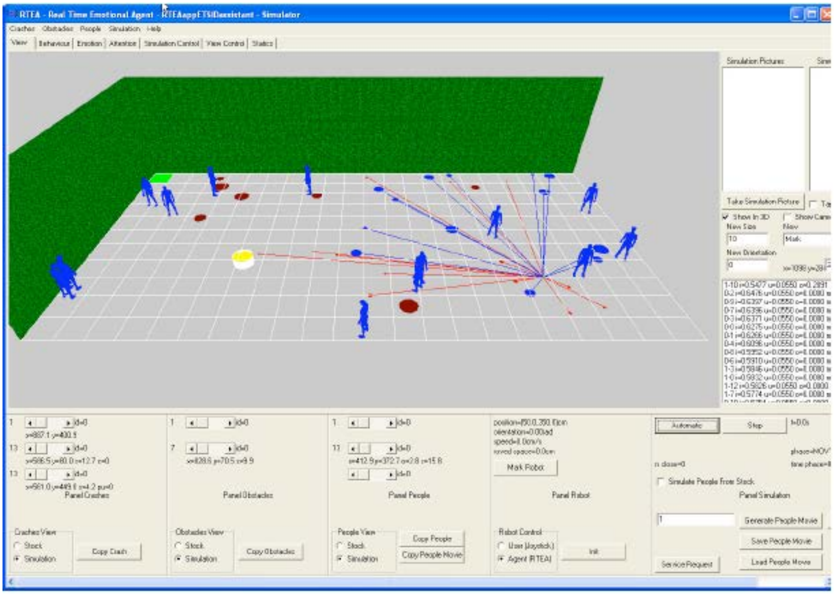

Fig. 6.7 Environment simulator - robot solving a crash in the operating area

6.4.2. Multicore implementation

In this section, the implementation of the parallelizable emotional processes is described. The emotional processes (see Fig. 6.4) detailed in section 6.3 are developed on the multicores. The emotional contributions are based on hyperbolic tangent functions as shown in Equation 6.4.

$$th(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{6.4}$$

In order to allow adjusting these emotional contributions, the function of Equation 6.4 is transformed in the function shown in Equation 6.5, where the parameters x_0, y_0, δ_x and δ_y serve to translate and scale the function.

$$th^{*}(x) = \left(\frac{e^{2(x-x_0)\delta x} - 1}{e^{2(x-x_0)\delta x} + 1} - y_0\right)\delta y \tag{6.5}$$

Regarding the implementation on a multicore, the robot architecture including the belief, behavior, attention and emotional sub-systems is implemented on a quad-core processor, the Intel Core i7-2600 at 3.4GHz per core. The i7 based computer has 8MB cache memory, 16 GB DDR3 RAM. One of the cores is dedicated to the application processes and the remaining 3 cores implement the emotional system. The number of active cores for the emotional system can be configured by the operating system, using the *sched_setaffinity()* system call in Linux. In the evaluations, the emotional processor is run in a single-core mode, dual-core and three-core, each with two hardware threads, to assess which is the sufficient number of cores to tackle the application problems. The processes are distributed evenly among the different cores. To this end, the Worst Fit algorithm is used to refill the cores [44]. The system model implemented is shown in Fig. 6.8.

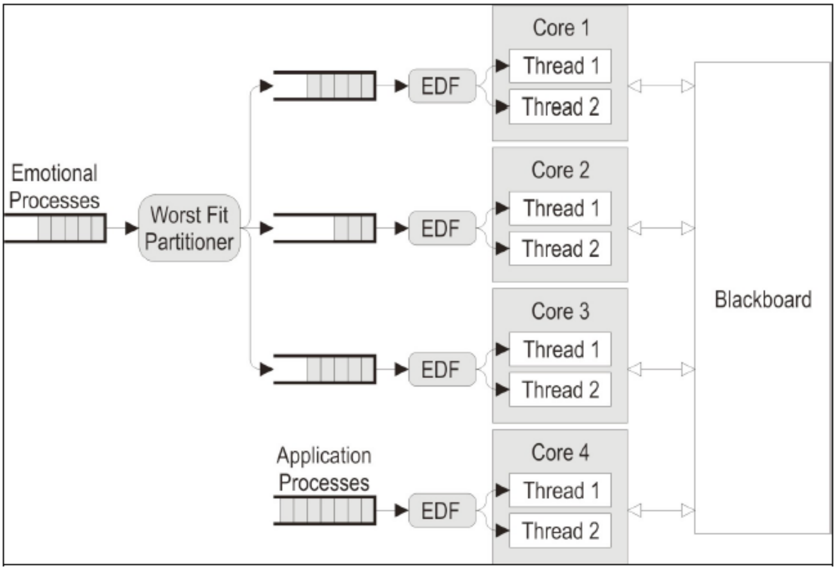

Fig. 6.8 Multicore architecture

Initially, the WF replenishes the core with the emotional processes. The policy of this strategy is to assign the process to the less loaded core until all the processes are assigned to their corresponding core. Each core implements an EDF Scheduler to execute the processes belonging to that core [41]. EDF prioritize the task that has the closest deadline. The schedulers and the processes are implemented as real-time tasks in the rt-linux kernel. The real-time processes are executed using the main memory without accessing the disk device.

The cores share a memory called the blackboard. Through this memory the processes read and write the important data as the appraisals, emotional states and motivations. Also it allows, using shared variables to synchronize the different processes. The attention core, at each attention cycle, activates the Belief processes that write in the blackboard the appraisal information. The emotional processes then read the appraisals and calculate the emotions to update the motivations in the memory. Finally, the behavior processes read the motivations and execute the actions prioritizing the behaviors with highest motivations. These operations are repeated periodically at each attention cycle.

6.5. Experimental evaluation

The robot executing different activities, under the aforementioned environmental, emotional state and dynamic conditions, is evaluated. The evaluation is focused on the analysis of the performance, measured on MOPS, that a multicore processor obtains running these applications using one, two and three cores with two hardware threads and at a frequency of 3.4GHz per core.

Firstly, the speed-up is calculated to compare the performances obtained by executing the same robotic applications using the different configurations of the processors. Fig. 6.9 shows the obtained speed-up, based on the MOPS running the emotional contribution functions of the Equation 6.5, on a single core - single thread processor, one core - two threads, two cores - four threads and three cores – six threads processors. As shown, speed-up improves as the number of cores and threads increase. Specifically, a one core - two threads runs 1.77 times faster than a single core – single thread, a two core - four threads 3.62 times and a three cores - six threads 5.33 times.

Fig. 5.10 to Fig. 5.12 show the results of evaluating the different implementation alternatives, considering 3 complexity problems (Simple, Normal and Complex) for 3 robot emotional state levels (Stressed, Normal and Relaxed).

In each figure, the bars represent the maximum computation capacity in MOPS that each processor provides (e.g., Fig. 6.10 shows that a three cores - six threads processor performs 144MOPS, a two cores - four threads 98MOPS, a one core - two threads 48MOPS, while a single core – single thread reaches 27MOPS).

The emotional states are represented with three different colors, being light grey the stressed state, dark grey the normal, black the relaxed state. These colors show at which emotional state the processor can carry out the problem. For instance, in the case of a simple problem, if the emotional state is relaxed the minimum computation capacity required by the processors is 51MOPS, while if it is stressed the required capacity is 11MOPS (see Fig. 6.10). This is because the attention cycle increases as the emotional state becomes more stressed, and then less computation MOPS are required. With a relaxed state, only two cores - four threads and three cores - six threads processor can solve this kind of problems, while in the other two emotional states all the considered processors tackle the problem.

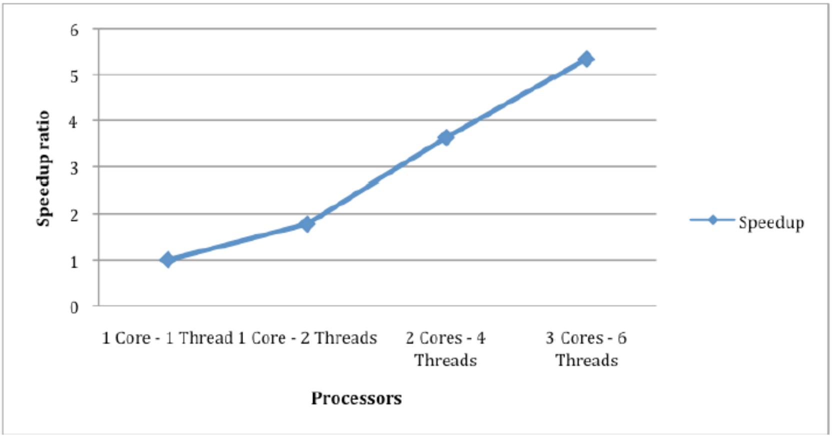

Fig. 6.9 Speedup comparison

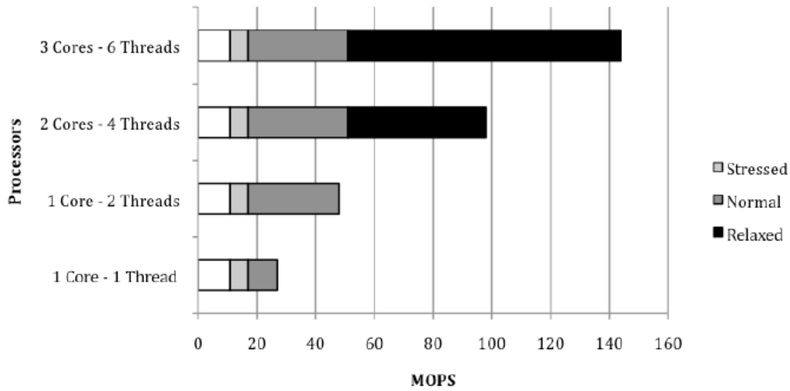

Fig. 6.10 Simple problem

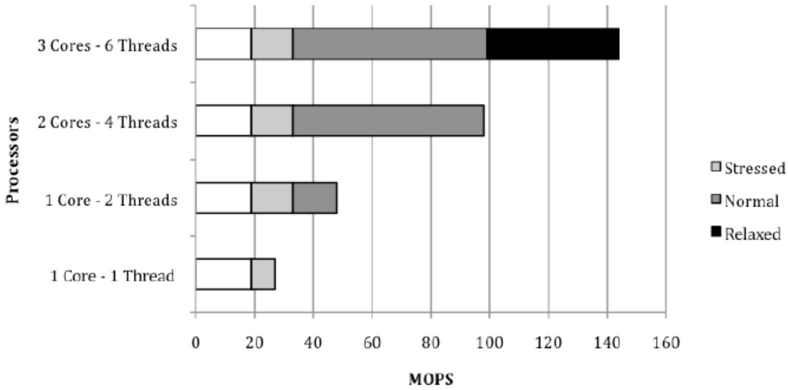

Fig. 6.11 Normal problem

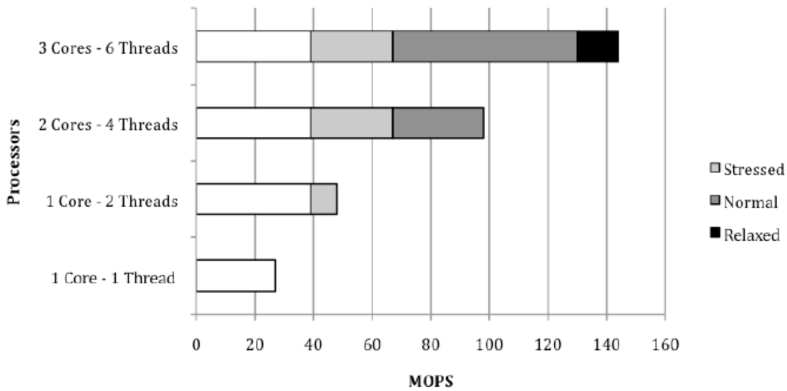

Fig. 6.12 Complex problem

On the other hand, as the complexity of the problem increases, the minimum computation requirements (MOPS) of the processors to solve the problem also increase. For instance, for a normal problem and relaxed robot the required MOPS are 99 (see Fig. 6.11), while a complex problem with the same relaxed robot, the 130MOPS are required (see Fig. 6.12).

Table 6.3 shows a comparison of the analyzed multicore processors in terms of the fulfilment of the computational requirements to solve the proposed robotic applications, considering the aforementioned application complexities and the robot stress levels.

This comparison allows identifying the processors that are able to execute the required computational workload for each case. In this sense, only the three cores – six threads processor can tackle complex problems at a relaxed state. Two cores - four threads is the next more powerful solution, hence suitable election to solve less constrained problems than the previous one but still complex ones (e.g., complex problem and normal robot). However, this processor fails tackling some normal problems with relaxed robots. A single-core with two threads never permits the robot to be in a relaxed emotional state. Finally, the single core - single thread processor can be used only for solving simple problems.

Table 6.3 Processor suitability for different problem complexities and robot emotional states

Problem	Robot state	MOPS	1Core 1Thread	1Core 2Threads	2Cores 4Threads	3Cores 6Threads	Processor
Simple	Stressed	11	√	√	√	√	
	Normal	17	√	√	√	√	
	Relaxed	51			√	√	
Normal	Stressed	19	√	√	√	√	
	Normal	33		√	√	√	
	Relaxed	99				√	
Complex	Stressed	39		√	√	√	
	Normal	67			√	√	
	Relaxed	130				√	
			27	48	98	144	Capacity (MOPS)

6.6. Conclusions

A multicore-based emotional control architecture to implement future mobile robots has been presented. The emotional processes of the architecture have high computational requirements, which consume the computational power of the main core. To reduce this consumption, the parallel capabilities of the emotional processes of the architecture have been exploited and the implementation of the emotional processes on high performance multicore processors has been tackled. An industrial mobile robotic application (under different environmental, dynamic and emotional robot state conditions), implementing the emotional based multicore architecture has been proposed. The robotic application performances have been evaluated for three-core and dual-core processors with up to six hardware cores. Results show that three-core with six threads can solve most complex problems under the most constrained requirements. Dual-core with four threads is the second more powerful solution hence suitable election to solve most part of the complex problems, however, it fails tackling normal problems with relaxed robots. A single-core with two threads never permits the robot to be in a relaxed emotional state. The single core can only solve the simplest problems.

Acknowledgment

This work was supported in part under Grant of the Universitat Politècnica de València number PAID-2012-325.

7. Publicación E:

Real-time Emotional Agent Architecture Application on Service Mobile Robot Control

Carlos Domínguez, Houcine Hassan, Alfons Crespo.

The 2009 International Conference on Artificial Intelligence.

Abstract

This paper presents an application of RTEA – Real Time Emotional Agent. RTEA architecture uses emotions to motivate the behavior. The agent is able to face multiple simultaneous activities in dynamic environments with real time requirements. The RTEA's behavior is problem based. In the solving problem process, the agent decomposes complex problems in simpler ones and generates a tree of desires. Every desire starts a thought that tries to get to the desired situation. A set of emotions organize the desire selection and therefore the effective behavior. An attention system negotiates the thoughts' dedication using an attention policy. The agent adjusts the problem conditions that are under its control to limit the risk of arising undesirable situations. The application presented in this article consists of a mobile robot that offers services. We have considered different emotional appraisals sets (agent characters) and different complexity levels for the navigation environment to evaluate the "emotional motivation" and the "controllable conditions adjust" mechanisms in the RTEA architecture.

7.1. Introduction

In this paper, we have defined an artificial agent's architecture - RTEA (Real Time Emotional Agent). RTEA uses emotions to motivate the behavior of the agents.

We had three kinds of design goals related to: (a) on what kinds of problems agents could help us, (b) what should be a convenient agent's profile (c) how should an agent's architecture contribute to build such agents.

About the problem

We deal daily with many simultaneous problems and do it with limited resources. These problems arise in dynamic and uncertain environments with real time requirements. Many of our decisions during the day are about what problems should we focus on, having to keep apart other unsolved important problems because of our limited capacities, and when we finally decide to go for a specific goal we usually realize that we are going to accomplish it partially. We are used to live with unsatisfied desires all the time and it is fine because we are not super heroes – we cannot have everything in life. However, when we decide to do something, we are supposed to act responsibly, because our behavior affects other people. Those are the kinds of problems we would like our artificial agents be able to deal with - the same problems that we face every day.

About the agent

There are some desirable characteristics for artificial agents. (1) They should be controllable (by assuming the desires of the user) and their behavior should be stable and predictable. In this work we wanted to test the controllability and stability of RTEAs – since they are complex emotional systems that could run out of control. The results of the experiments in this work show that the controllability and stability are possible. (2) The behavior should be effective. That depends on the quality of the methods that the agent developer defines for the specific problems and the goals selection. The results of the experiments show how the different agent attitudes affect the effective behavior and therefore the satisfaction of the final reached situations. (3) The behavior should be responsible. The experiments also show how the mechanism that adjusts the controllable conditions (e.g. adjusting the robot speed to maintain the collision risk) in RTEA works. (4) The agent should learn. In RTEA there are two kinds of knowledge that must be learned and improved – they are related to the procedural intelligence (the solving methods for the specific problem domain) and the emotional intelligence. (5) We want that the agent be autonomous in all the former behaviors.

About the architecture

Agent architectures should guide developers in the building agent process. A guide means also constraints, but they should be only those strictly necessary. In the case of RTEA, the main contribution is to help in the organization of the desire and behavior processes. The specific problem solving processes have to be developed by experts in the corresponding fields. In order to let those processes to be embedded into the agent mind, the developer must implement the architecture interfaces. That is the cost.

The paper is organized as follows: In section 6.2 the related work is discussed. Section 6.3 presents the key elements of the architecture. Section 6.4 describes an application example, which is based on a service mobile robot. Two experiments have been developed to test the "emotional motivation" and the "controllable conditions adjustment" mechanisms in RTEA. Finally, section 6.5 summarizes the results.

7.2. Related work

An ethological model based on the praying mantis was proposed by Arkin et Al. [46]. In this model the robot maintains three motivational variables that represent the robot's hunger, fear and sex-drive. These internal variables are modeled linearly with time. Action selection is used to enable the behavior associated with the motivational variable with largest value.

Sony in collaboration with Ronald Arkin developed a computational model based on canine ethology and was used in the design of the Aibo entertainment robot [47]. The model consists of six basic emotional states: happiness, anger, sadness, fear, surprise, and disgust. Each of them is reduced into three dimensions: pleasantness, arousal, and confidence. The levels of the internal variables are established and by relating the robot state with these thresholds the emotional state can be assessed. The resulting emotional state affects the action-selection process in the behavior eligibility. Based on the motivational space and action-selection mechanisms behaviors are scheduled for execution.

Moshkina has proposed the affect model Tame in order to assist in creating better human-robot interaction [48]. The model captures the interaction between different affect phenomena, such as traits, attitudes, moods, and emotions. In Tame, emotions are high activation and short term, while moods are low activation and relatively prolonged. Traits and attitudes determine the robot disposition and are time invariant. The affect model gets perceptual information and modifies the underlying behavioral parameters, which in turn, directly affect currently active behaviors.

Cynthia Breazeal developed one of the first social robots: Kismet [49]. Kismet's motivation system consists of drives and emotions. The affective space is defined by three dimensions: arousal, valence and stance. The emotion is computed as a combination of contributions from drives, behaviors, and perceptions. The motivation is taken into account in the behavior selection and in the activation of facial emotional expressions.

Arbib proposed a robot composed of a set of basic functions, each with a perceptual schema and access to various motor schemas [50]. The perceptual schema evaluates the current state and sets an urgency level parameter for activating the motor schemas. A motivational system is defined to adjust the relative weighting of the different functions, raising the urgency level for one system while lowering the motivation system for others, depending on the context.

In Malfaz [56], an emotion-based architecture has been proposed. In this architecture emotions are generated from the evaluation of the wellbeing of the robot. In general, the research conducted in the topic of emotional systems has been focused in the human-robot interaction [46, 47, 49]. The work presented in this paper focuses in the emotion as a mechanism of the organization of the robot behavior. In this area, conceptual frameworks that allow to express control concepts and higher level mental processes found in biological systems to help understand the kind of situations where emotional control mechanisms, are claimed to be employed [50, 51], more realistic models to analyze how changes the agent's emotional state based on new percepts [52], and more accurate model of the complex appraisal processes and emotion effects on decision-making are also research challenges in this topic [53].

7.3. Architecture

7.3.1. Global overview

RTEA's behavior is problem oriented. A problem arises when the agent desires to change a situation. When a new desire is created a thought that tries to satisfy it is started immediately. The thought defines a method (procedural intelligence of the agent) to solve the situation change problem. Most of the times, the method decomposes the main problem and generates a set of simpler problems. Then, every new problem is represented by a new desire-thought pair. At a specific point the RTEA can have a large tree of desires – being the root node the initial desire (usually user defined) and the descendant nodes the desires generated by the agent in the problem solving processes.

Since the mental capacity of the agent is limited, some of those desires must be taken apart for a while. So, on which of those desires should the agent focus on? The emotional system of the agent has the answer. Every thought has an emotion which motivates it. This emotion, named "intrinsic emotion" because it is related to the thinking process situation instead of the problem (application) situation, appraises some important parameters related with the expected satisfaction that the agent could get if it is dedicated to this specific problem, and then, sets a motivation level for the thought. If a desired situation is considered important (because the solving problem method says that it is a profitable situation) then the initial motivation of the agent takes into account that level of importance. But maybe the actual conditions don't let the agent to get the desired situation and then the agent shouldn't waste his resources right now. Here is where the intrinsic emotion modulates the motivation.

Most of the appraisals related with the expected satisfaction consider things like: the confidence on the initial perceptions or observations (that define the problem), the confidence on the ability of the agent to handle the problem (method), the confidence on the availability of necessary device motors (if they have to act in an exclusive way) etc. Depending on the specific set of emotional appraisals that are used to motivate the thoughts, we define different agent characters. In the experiments that we present in this article we have considered two additional "motivation modulator" appraisals: the urgency and the opportunity. It is easy to observe these modulators in natural agents' behavior.

On the one hand, when some desired situation becomes urgent, it is common to give it attention, even though there are other more important desires in the agenda. It is as if the urgency distorts the real importance of the things. On the other hand, in a dynamic multi-problem context, it is possible to find opportunities. That happens when a desired situation (maybe not so important) seems to be easy to get from the actual situation and then the agent decides to go for it trying to get a short-term reward.

These are examples of the kind of behavior regulators that RTEAs can implement and the “emotional motivation mechanism” of RTEA architecture is the system that handles them.

Once a problem is chosen from the agenda a set of observation-decision-action processes that work on the situation transformation must be given attention. In order to not exhaust the agent mental capacity and let other problems to be considered at the same time it is necessary to set an attention level. The attention system negotiates the dedication of every thought and the key parameter in this negotiation is the motivation. It is possible to define different attention policies and then to get different agent characters (e.g. agents that prefer to attend small sets of complicated problems at the same time, or agents that prefer to deal with many simpler problems, etc.)

When a thought gets a level of attention from the attention system, the controllable conditions of the situation transformation must be chosen in a responsible way. If not, undesirable situations (e.g. dangerous or expensive situations) could happen. The responsibility in RTEA is defined based on the concept of risk. The agent in the experiments shown in this article, for example, sets the maximum robot navigation speed depending on the actual navigation complexity (e.g. number of mobile obstacles around and their relative speeds) and the acceptable risk of a collision situation and its effects. Again, different acceptable risk levels lead to different agent characters, to different behaviors and eventually to different final situations. The “controllable conditions adjusting mechanism” of RTEA handles the responsibility. In order to be able to do that, the developer must provide some explicit models in each desire-thought context: a “situation satisfaction model”, a “dedication model” and a “situation change model”. They are part of the interface that the developer must implement in order to embed their situation transformation methods in the RTEA architecture. These models make explicit some information that usually is implicit in other (not agent based) controller implementations. If that information is presented explicitly then the agent architecture can help the developer on the behavior organization.

7.3.2. Main processes and data flow

Fig. 7.1 shows the main processes and concepts in RTEA. There are three main ways in the data flow.

The reactive way connects sensors to motors through reaction processes. A reactive process is based on a method with well-known time cost. Reactive behavior is necessary to prevent undesired (unsafe or expensive) situations.

The deliberative way connects sensors to motors through deliberative processes (observations, deductions, decisions and actions). A deliberative process is based on a method with no fully predictable time cost. The final time depends on events that happen during the process. Deliberative behavior is necessary to handle most of the real problems because of the intrinsic environment uncertainty.

In the context of every problem (represented with a desire-thought pair), reactive and deliberative processes coexist.

The emotional way implements the motivation and attention processes. It is actually a reactive way, because the emotional processes are based on well known time cost methods, those that guarantee that the behavior organization is under control.

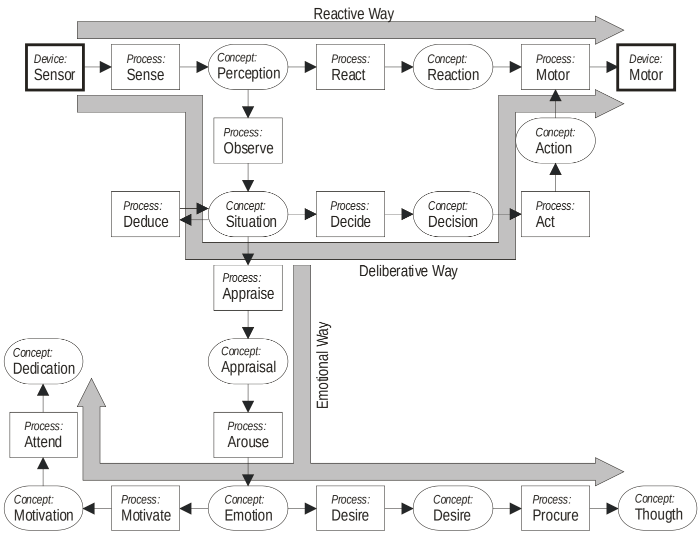

Fig. 7.1 RTEA data flow

7.3.3. Implementation

RTEA is implemented in 5 main subsystems: belief, emotion, behavior, attention and relation.

The belief system maintains a logical image of the environment. The processes in execution read and update this image permanently. The emotional system is the motor of the mental organization. It manages a set of emotions to set the motivation of each of the active thoughts. The behavioral system defines the behavior of the agent. The main entity of this system is the thought, a kind of conscious process. The attention system organizes the execution of the processes. This system negotiates with the thoughts in order to get relevant information to guarantee their execution (security requirements) and to determine the degree of satisfaction of their desires (functional requirements). Finally, the relation system communicates the agent with its environment.

The RTEA architecture has been implemented in a real-time kernel (rt-linux) that is very suitable for real-time control applications [55].

7.4. Application and evaluation

We have considered an assistant agent based on a mobile robot platform. The agent is able to perform some basic services - transport of small objects, cleaning, etc. We have evaluated it using a simulator. See Fig. 7.2.

The main evaluated aspects have been:

1. The emotional mechanism that motivate the behavior.
2. The mechanism to adjust the controllable conditions depending on the complexity of the problem.

We have defined two separated experiments.

Experiment A - *The agent deals with a global problem in a non-dynamic environment. The controllable conditions (e.g. navigation speed) are constant.*

The agent solves the problem by decomposing it in a set of smaller problems. The emotional system has influence in the decision taken in the selection of the specific problem. Depending on the agent character, his attitude facing each specific problem changes during the process of solving the global problem, due to this fact the agenda becomes a dynamic list. Different execution sequences give different final satisfaction level for the global problem. The experiment shows how much sensitive is the final satisfaction level to the attitude of the agent.

Experiment B – *This is the same problem as before but in a dynamic environment. In this case, the agent must adjust the controllable conditions (e.g. navigation speed) in order to maintain the risk level of some undesired situations (e.g. mobile obstacle collision).*

Different environment conditions (e.g. distances, relative speeds, etc.) have different process cost (e.g. observation of the people moving around the robot, planning of a suitable trajectory, etc.) The attention system negotiates with the mental processes their dedication dynamically and every process must adjust the controllable conditions to maintain the risk level under a maximum limit value.

7.4.1. Experiment definition

Firstly, the problem to solve and the environment are presented. Secondly, the specific characteristics of the experiments will be set-up.

The agent is able to perform two kinds of maintenance services: “to clean dust spots” and “to pick up pieces”. Additionally, the robot is able to perform generic operations that are necessary to offer those services: “to observe the local environment using a camera”, “to plan trajectories leading to specific places”, “to act and follow planned trajectories” and “to react avoiding mobile obstacle collisions”.

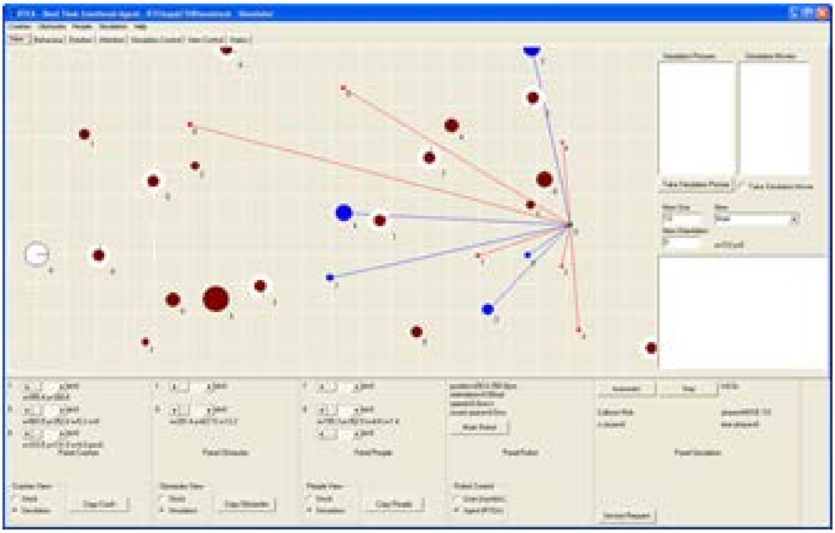

Fig. 7.2 Simulation framework

7.4.1.1. The environment – navigational space

The navigation space has been limited to: XMin=0cm, XMax=1100cm, YMin=0cm and YMax=550cm

7.4.1.2. The environment – physical objects

The environment contains two kinds of objects that interfere with the robot navigation: “Static Obstacles” and “Mobile Obstacles”. They all are represented with their position and size. Additionally, the mobile obstacles are defined with their orientation and speed. The simulator defines the behavior of the mobile obstacles; they move around the navigation space and interfere with the robot navigation in a no completely predictable way.

7.4.1.3. The problem

The agent must fix eventual crashes. When a crash happens, a set of pieces and a set of dust spots are spread around the floor. A new problem for the agent (service request from the user) is defined as follows:

- The crash to fix.
- The importance of the service - used to motivate this service against other eventual services in the agenda.
- The satisfaction temporal model - that sets the deadlines in order to evaluate the service outcome from a maximum satisfaction level to a maximum frustration level.

When the agent sets a new “Fix Crash” desire based on a service request, a “Fix Crash” thought is started. Then the method used to fix crashes decomposes the problem in simpler parts, so that a set of new desires are settled by the agent in an autonomous way. Every new desire focuses on a part of the main (crash) problem - on a dust cleaning or on a piece picking up simpler problem. The decomposition method defines the importance and the temporal satisfaction model for every secondary desire accordingly to the level of inconvenience of those pieces or spots.

7.4.1.4. The robot

To fix the problem, the agent controls a mobile robot platform.

7.4.2. Definition of the specific experiences

7.4.2.1. Experiment A

The agent has to fix a crash in an environment without mobile obstacles. The agent has been tested for 4 different characters. See Table 7.1

Table 7.1 Agent characters

Character	Motivation Contributors
I	Importance
I+U	Importance + Urgency
I+O	Importance + Opportunity
I+U+O	Importance + Urgency + Opportunity

Every character has different sets of emotional contributions to motivate the behaviors. The contributions considered in this experiment come from the following situation's appraisals:

- Importance – Importance of the desired situation.
- Urgency – Relative temporal difference from the expected finalization time and the desired time.
- Opportunity – Relative expected cost (in this case: time of robot use) for the expected satisfaction.

The emotional appraisals use the estimated times for operation (cleaning dust or picking up piece) and for navigation. The cleaning operation estimated time is proportional to the dust spot size. The picking up operation estimated time is constant. The navigation time is estimated using the robot actual speed and the number of mobile obstacles in the area. In order to compare the different attitudes we have considered a set of 10 different scenarios. The problems (crashes) have been randomly defined from very simple ones to very complex (from 0 to 24 dust spot / piece) in increments of 5 objects steps.

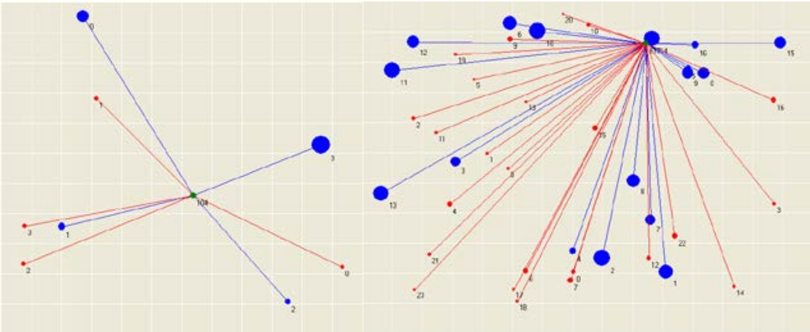

Fig. 7.3 Different complexity scenarios

7.4.2.2. Experiment B

The agent has to fix a crash in an environment with mobile obstacles. The agent has been tested with the attitudes shown in Table 7.2. The different attitudes come from the characters defined in the experiment A and from the 3 levels of risk that have been assumed (low, medium and high). The actual risk level is adjusted through the controllable conditions (e.g. navigation speed).

Table 7.2 Robot attitudes

Anima Character	**Low risk**	**Medium risk**	**High risk**
I	I-low	I-med	I-high
I+U	I+U-low	I+U-med	I+U-high
I+O	I+O-low	I+O-med	I+O-high
I+U+O	I+U+O-low	I+U+O-med	I+U+O-high

In order to force the agent to modify the controllable conditions we have defined different sets of moving people in the crashes scenarios of the experience B. The different people scenarios have been defined for 5, 10, 15 and 20 people with random evolutions (positions, orientations and speeds).

7.4.3. Definition of the analysis

In these experiments, we want to observe the emotional mechanisms in the agent architecture running.

1) The motivation and behavior selection mechanism.

We use the experiment A to compare the different agent attitudes (importance, urgency and opportunity appraisals). We compare the execution sequences and their effect in the satisfaction level of the final situation. We observe the total time for the operation.

2) The controllable conditions adjusting mechanism.

We use the experiment B to compare the different agent attitudes (level of assumed risk) and the lack of adjusting mechanism. We compare the execution sequences and their effect in the satisfaction level of the final situation. We observe the total time of the operation and the number of collisions.

7.4.4. Experimental results

For each of the experiments A and B we present the results in two parts: (1) we explain a simple simulation sequence (selected from the bank of simulations) in order to remark the key events (e.g., motivation inversions and mind changes, collision appraisals and speed adjust, etc.) (2) we analyze statistic results from the bank of simulations.

7.4.4.1. Simple execution sequence for experiment A

We have chosen a simple problem represented in Fig. 7.4

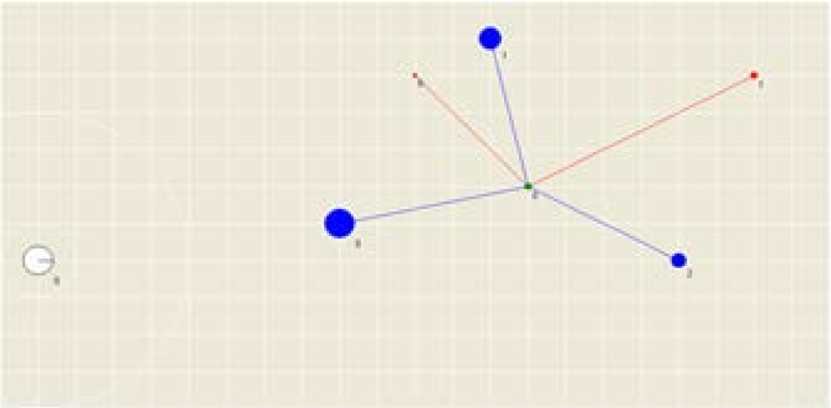

Fig. 7.4 Simple problem example

After the global problem decomposition, the agent has 5 small problems in his agenda. It sets the importance and the deadlines depending on how much every problem affects the navigation. The bigger dust spots or pieces size are the more is the navigation interference. Piece 1 is considered to have the shortest deadline because it is placed next to an important access to the navigation space. The deadline column represents the deadline for maximum satisfaction and the deadline for the null satisfaction (being the satisfaction model a ramp based function).

Table 7.3 Subproblems

Problem	X(cm)	Y(cm)	Size(cm)	Importance	Deadline(s)
Dust0	450	300	20	0.700	180..240
Dust1	650	50	15	0.650	180..240
Dust2	900	350	10	0.600	180..240
Piece0	550	100	3	0.560	180..240
Piece1	1000	100	5	0.600	90..120

Character 1 – Importance

If the agent only appraises the importance of the problems then it motivates them in the following order: Dust0, Dust1, Dust2, Piece1 and Piece0. The robot invariably follows that constant motivation list when it selects the next problem to fix.

The global problem is fixed after 134.6s. Fig. 7.5 shows the marks of arrival time over every specific problem place.

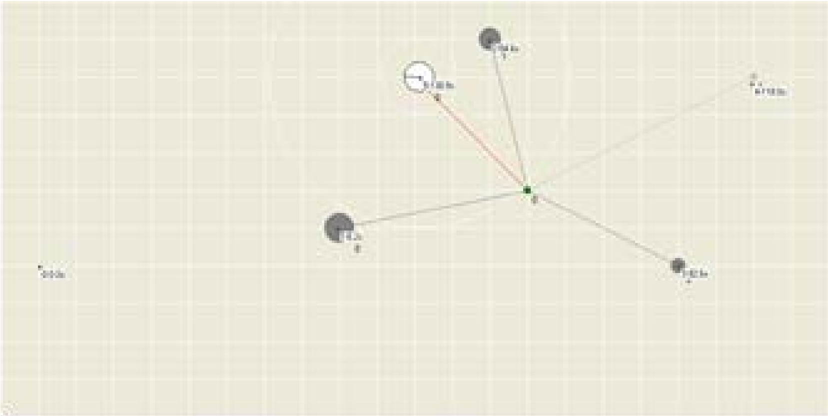

Fig. 7.5 Sequence for only importance appraisal

Character 2 – Importance and Opportunity

If the agent appraises the importance of the problems and the opportune situations, then the motivation list becomes variable and the fixing problem sequence changes during the experiment.

The global problem is fixed now after 126.2s because of Piece0 has been fixed at some point in the way of the robot to fix Dust1. The time when the agent changes his mind can be observed in mark 2 (at 51.4s) in Fig. 7.6.

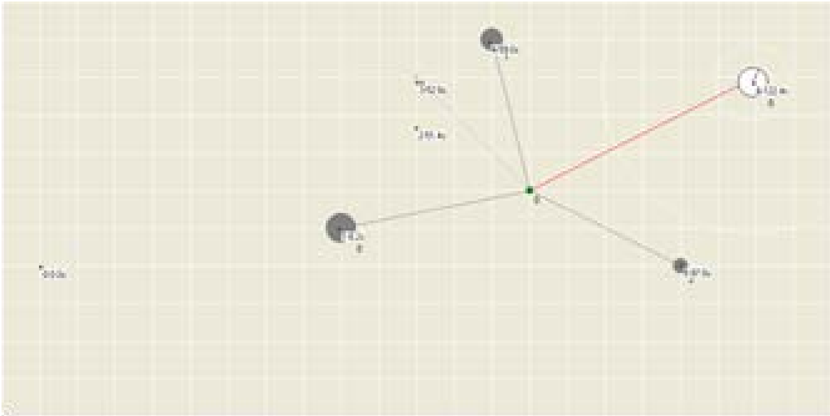

Fig. 7.6 Sequence for importance and opportunity appraisals

Character 3 – Importance, Urgency and Opportunity

If the agent appraises the importance of the problems, how urgent they are, and the opportune situations, then the sequence is different again. In this example the robot changes his mind when Piece1 problem becomes urgent. See mark 5 (at 90.2s) in Fig. 7.7. The agent changes its objective of preventing to get into the frustration interval time (120s, $\propto$) of the satisfaction model for Piece1. It decides to suspend temporally the actual problem process (Dust 2).

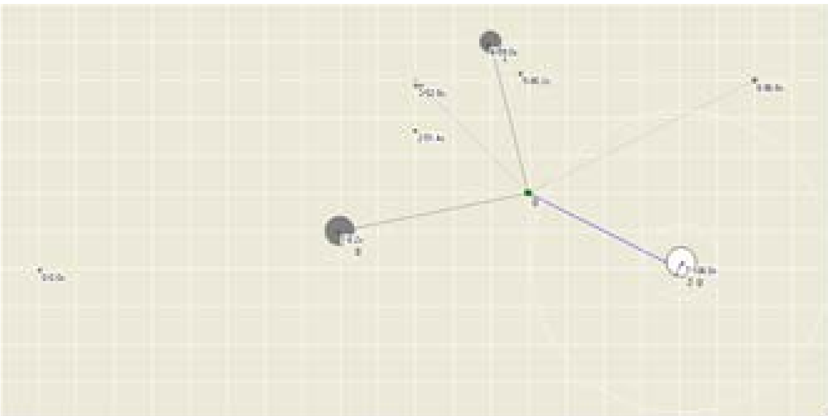

Fig. 7.7 Sequence for importance, urgency and opportunity

7.4.4.2. Simple execution sequence for experiment B

Adjustable speed and medium risk

Fig. 7.8 shows the marks of a simulation sequence in an environment with people moving around.

If the agent is able to adjust the navigation speed to the actual conditions then it is possible to navigate quickly through clear spaces and slowing down speed when the conditions become complicated. For a medium risk level and for only importance appraisals, the agent finished the operation after 146.4s.

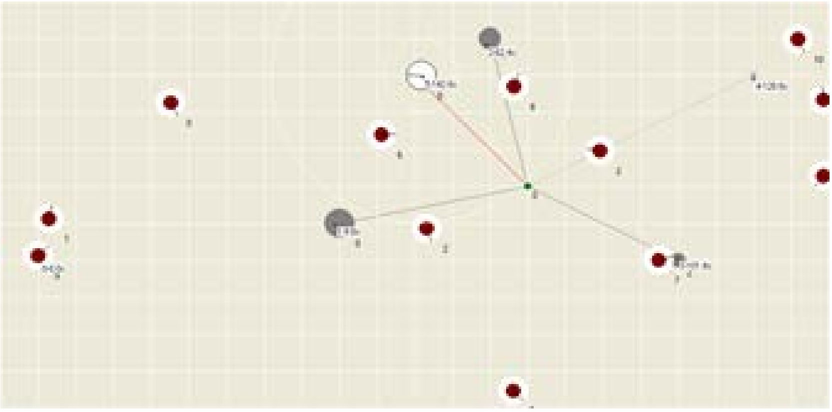

Fig. 7.8 Final situation with adjustable conditions

Constant speed and medium risk

If the controllable conditions adjust mechanism is turned off, then the agent must navigate with a constant speed, which must be compatible with the maximum number of mobile obstacles expected during the whole fixing operation and the admissible collision risk level. In that case, for the same environment conditions the agent finished the operation after 277.1s.

7.4.4.3. Statistics results for experiment A

Fig. 7.9 shows the final time of the operation for different crash scenarios (number of secondary problems) and different agent characters.

The final time to fix the crash increases with the number of parts in the crash. We observe however that it increases quite lineally. Perhaps more lineally than it should be expected for this kind of problems where the spreading area for the pieces and dust spots is limited (and the probability to find close pieces/spots should increase with the number of parts). This is due to the parameters that we used to define the problems, where the navigation time contributes significantly less than the operation time. It can be observed that the contribution of the opportunity appraisal, in both of the characters that consider this appraisal: I+O and I+U+O, makes the final time to fix the crash be reduced.

The appraisal of urgency has contributed negatively in the final time. We can observe this in both of the characters that consider the urgency: I+U and I+U+O

In these cases the agent was trying to minimize the frustration caused for missing deadlines instead of simply minimizing the final time.

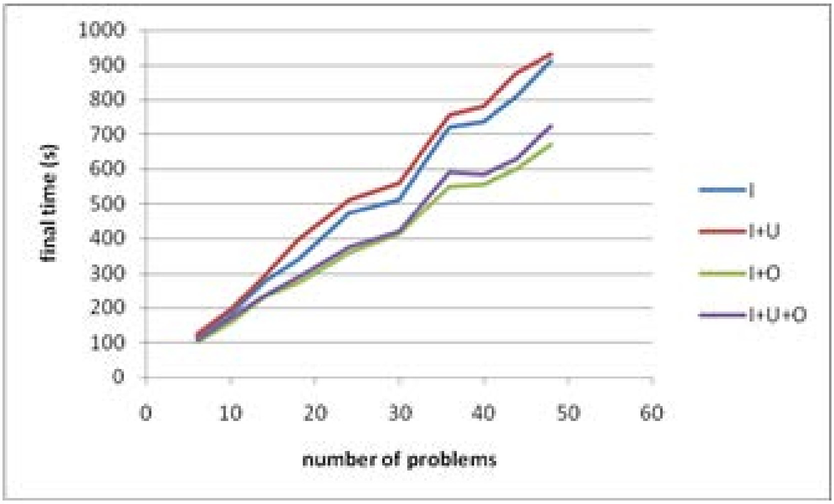

Fig. 7.9 Final time for the bank simulations of experiment A

7.4.4.4. Statistics results for experiment B

Fig. 7.10 shows the final time of operations for different crashes and people scenarios (number of secondary problems and number of mobile obstacles) for importance appraisal and medium risk level when the speed adjust mechanism is on.

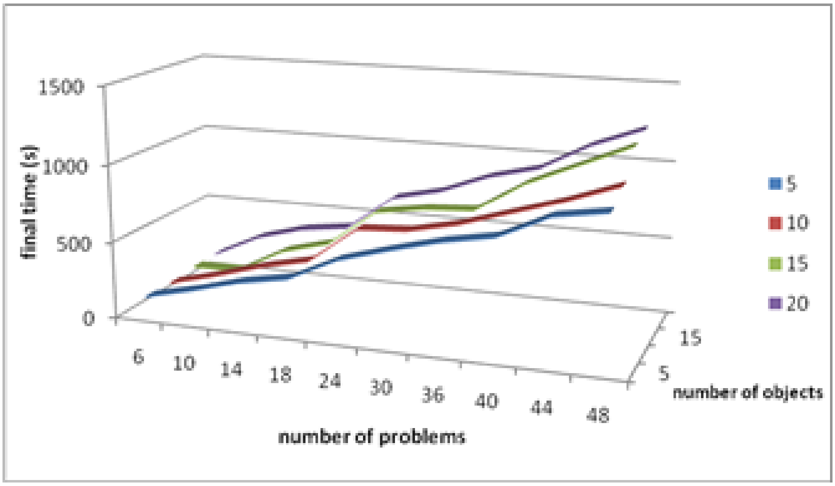

Fig. 7.10 Final time for experiment B regulable speed

Fig. 7.11 shows the final time of operations for the same crashes and people scenarios and the same attitude than Fig. 7.10 but with the speed adjust mechanism turned off. Observe the different final time scale in the figure. The constant navigation speeds of the robot have been: 5cm/s for 20 people around, 10cm/s for 15 people, 20cm/s for 10 people and 40cm/s for 5 people. The reaction behavior to prevent collisions was active.

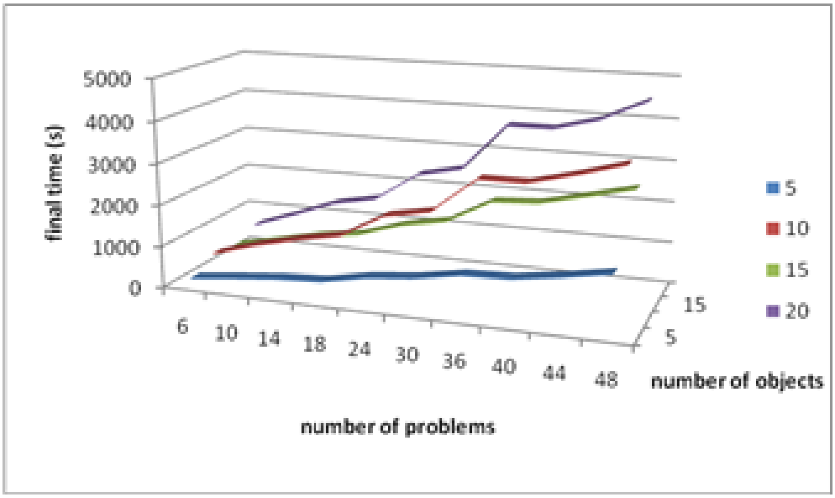

Fig. 7.11 Final time for experiment B constant speed

7.5. Conclusions

This experiment has shown two of the emotional mechanisms in RTEA architecture and how the attitude of the agent can drastically modify the effective behavior of the agent. These changes in the behavior can lead in significant differences in the satisfaction level of the final situation. So, the emotional parameters can be used as an important control behavior mechanism and this is our main motivation to keep researching in emotional agent architectures.

8. Resumen de Resultados y Conclusiones

En esta tesis se ha definido una arquitectura de agente emocional, en la que el proceso de motivación y atención es dirigido por una versión simplificada de la emoción en los agentes naturales. Se ha propuesto un mecanismo de motivación y atención, que constituye el proceso operativo del agente.

El comportamiento del agente se organiza en torno al concepto de problema y al proceso de resolución del mismo. La emoción motiva los problemas dirigiendo la atención del agente hacia los más prometedores.

Efectividad del mecanismo de motivación y atención emocional

Mediante una aplicación de robot móvil de servicio se ha evaluado el mecanismo de motivación y atención. Se ha comprobado que se trata de un mecanismo efectivo, ya que permite la selección de comportamientos utilizando como criterios, apreciaciones subjetivas sobre la importancia del problema a resolver y su expectativa de éxito de resolución. Con la definición de diferentes caracteres de agente, lo que supone la selección y configuración de diferentes grupos de contribuciones emocionales, se ha comprobado que es posible adaptar el agente a problemas de aplicación específicos.

Ventajas para el diseño de aplicaciones de agentes emocionales

El constructor de agentes necesita definir las interfaces que el mecanismo de motivación y atención emocional requiere para cada comportamiento que se desee integrar en el agente. Esto supone tener que definir los *Modelos Explícitos* de *Satisfacción*, *Dinámica* y *Dedicación* para cada uno de los problemas de la aplicación. Sin embargo, pese a este esfuerzo de diseño, el constructor de agentes obtiene como recompensa, un proceso de atención automático pre-implementado, que tiene en cuenta las distintas necesidades de dedicación de cada problema y la limitación de los recursos de procesamiento. El marco de diseño ayuda en el proceso de construcción del agente, que resulta más simple al estar los procesos operativos explícitamente separados de los procesos de aplicación.

Evaluación del coste de proceso

El proceso de motivación y atención emocional tiene altos requerimientos de cómputo, ya que debe considerar el conjunto de problemas activos en la agenda del agente, y no sólo los problemas que reciben atención. Se ha considerado el impacto que el proceso emocional tiene en el procesador principal y se ha planteado distintas alternativas para reducirlo.

Para poder evaluar el coste del proceso de motivación y atención emocional, se ha considerado distintos niveles de complejidad del problema (número de operaciones) así como distintos niveles de carga emocional (una medida relativa del coste del proceso operativo de motivación y atención respecto al coste de los procesos de aplicación práctica que resuelven los problemas). La elección de los niveles de coste se ha estimado para aplicaciones de robótica móvil de servicio. Se han considerado costes para distintos escenarios, considerando el número de problemas a resolver, que afecta al número de cálculos, y la dinámica de los problemas a resolver, que afecta al tiempo de respuesta requerido.

Evaluación de la carga de proceso

Con objeto de diseñar el procesador que permita ejecutar los procesos emocionales se ha considerado la carga emocional del agente. Se han establecido límites prácticos para dicha carga emocional que no debería sobrepasar un valor máximo bajo el riesgo de que el agente quedase sobrecargado emocionalmente y que no fuese capaz de solucionar problemas prácticos.

Para limitar el nivel de carga emocional, en un escenario con una complejidad dada, se ha establecido las prestaciones requeridas del procesador, medidas en operaciones emocionales por unidad de tiempo.

Alternativas de aceleración del proceso emocional para minimizar su impacto en el procesador

Se deseaba acelerar la ejecución del proceso emocional para permitir tratar problemas dinámicos que requieren tiempos de respuesta más breves, a la vez que poder tratar problemas más complejos, que se descomponen en un mayor número de subproblemas, a la vez que limitar la carga emocional a un límite práctico aceptable. La ejecución de los procesos de motivación emocional de cada uno de los problemas en la agenda es altamente paralelizable, debido a que pueden considerarse procesos independientes en cada ciclo de atención.

Se ha decidido implementar un coprocesador para ejecutar el proceso de motivación y atención, y se han considerado distintas alternativas disponibles: procesadores multicore, aceleradores gráficos GPU, uso de instrucciones SIMD, o el diseño de un coprocesador específico sobre FPGA.

Se ha comparado dichos niveles de carga con las capacidades de procesamiento que las distintas alternativas de coprocesador emocional planteadas con objeto de tener una lista de selección de procesador dependiendo del problema a resolver.

Multicores

Los procesadores actuales disponen normalmente de más de un núcleo de ejecución. Una alternativa de aceleración que supone una distribución de carga sencilla, ha sido utilizar hilos de ejecución concurrentes para la ejecución del código potencialmente paralelizable de los procesos de motivación y atención. Se ha aplicado OpenMP, modelo de programación para procesadores con memoria compartida.

Ha sido posible dedicar uno o varios núcleos para la ejecución de los procesos emocionales. Las prestaciones aumentan casi proporcionalmente al número de cores, con una ligera pérdida por el acceso a memoria compartida.

Instrucciones SIMD

Por otra parte, las instrucciones SIMD están también disponibles en las arquitecturas de procesador actuales, con lo que pueden acelerar la ejecución del proceso de motivación emocional sin coste adicional, tanto en sistemas monoprocesador como multiprocesador.

Se han probado instrucciones de aceleración basadas en el estándar SSE y AVX.

El coste de recursos adicionales es nulo, pues el juego de instrucciones está disponible en el procesador. Las aceleraciones se han podido obtener tanto para la implementación del agente original con un solo procesador como para la implementación en multicore, pudiéndose combinar el modelo de programación OpenMP con las instrucciones SIMD.

GPUs

Los dispositivos de aceleración gráfica también son comunes en los computadores actuales. Las GPU disponen de una gran capacidad de cómputo, en la forma de múltiples núcleos y la posibilidad de definir múltiples hilos de ejecución, que se adapta bien a los múltiples procesos de motivación que requiere ejecutar el agente emocional. Se ha utilizado el modelo de programación Cuda-C de nVidia.

El impacto del proceso de transmisión de la información entre la memoria del procesador principal y la memoria del dispositivo de aceleración gráfica puede reducirse mediante: (1) la selección del subconjunto de problemas de la agenda que deben ser motivados en base a su cambio de estado entre ciclos de atención, lo que supone actuar al nivel de la estructura de pizarra del módulo de creencias del agente; (2) el solapamiento del proceso de cómputo emocional con el de transmisión entre espacios de memoria, lo que supone actuar sobre el bucle del proceso principal de atención; (3) la utilización de los distintos tipos de memoria en la GPU y la memoria mapeada en el espacio de memoria del procesador anfitrión.

El coste económico de la GPU debe tenerse en cuenta, sin embargo las aceleradoras gráficas están disponibles por defecto en cada vez más ordenadores actuales. El mercado de dispositivos de procesamiento gráfico ha reducido sus costes debido a la potente industria de videojuegos, permitiendo que se haya desarrollado el modelo de programación GPGPU – *General-Purpose Computing on Graphics Processing Units*, y sea una alternativa de procesamiento muy común en los centros de cálculo de altas prestaciones actuales.

FPGAs

Finalmente, la implementación de un coprocesador específico, ha supuesto utilizar recursos que normalmente no forman parte de los computadores actuales. El coste económico de una tarjeta de procesamiento basada en FPGAs debe tenerse en cuenta.

El diseño propuesto en este trabajo se ha basado en la definición de una pipeline para el procesamiento de las emociones y la síntesis de la misma en dispositivos FPGA Stratix de Altera. Según los recursos de las FPGAs concretas que se han utilizado, y que dependen de la familia de FPGAs, se han podido alcanzar frecuencias de funcionamiento distintas. El coste de la comunicación entre el coprocesador y el procesador principal debe tenerse en cuenta, y suele ser el parámetro más limitante, con lo que es importante el tipo de interfaces de comunicación que estén disponibles en forma de IPs - *Intellectual Property Cores* - para el dispositivo FPGA seleccionado.

Trabajo futuro

Como continuación del trabajo realizado en este proyecto se plantea las siguientes líneas:

- Sobre el mecanismo de motivación y atención emocional - Mejorar el proceso de aprendizaje emocional, de forma el agente ajuste su actitud de una forma más dirigida por la experiencia real del agente sobre la dinámica del problema, de forma que los cambios de actitud puedan finalmente mejorar el desempeño del agente en dicho dominio de problema.
- Sobre el marco que ofrece la arquitectura al constructor de agentes - Desarrollar un banco de desarrollo que facilite el proceso de diseño y construcción.
- Sobre las aplicaciones del agente emocional - Ampliar los tipos de problemas que el robot móvil de servicio pueda tratar, y considerar otros problemas de distinta naturaleza, para estudiar la adecuación del mecanismo de motivación emocional a los mismos.

Resumen de Publicaciones

Publicaciones Seleccionadas

Publicación A:

Multicore and FPGA Implementations

of Emotional Based Agent Architectures

Carlos Domínguez, Houcine Hassan, Alfons Crespo, José Albaladejo.

Journal of Supercomputing. 71(2) 479-507

DOI:10.1007/s11227-014-1307-6

Publicación B:

Embedded GPU and Multicore Processors

for Emotional-based Mobile Robotic Agents

Francisco Almenar, Carlos Domínguez, Juan-Miguel Martínez, Houcine Hassan, Pedro López.

Future Generation Computer Systems. 56(March 2016) 192-201

DOI: 10.1016/j.future.2015.05.010

Publicación C:

Embedded Multicore Processors

and SIMD Instructions

for Emotional-based Mobile Robotic Agents

Francisco Almenar, Carlos Domínguez, Juan-Miguel Martínez, Houcine Hassan, Pedro López.

2014 IEEE International Conference on High Performance Computing and Communications

ISBN-13: 978-1-4799-6123-8

Publicación D:

Multicore Parallel Implementation of Agent Emotional Processes

Carlos Domínguez, Houcine Hassan, Vicent Mayans, Alfons Crespo.

2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications

DOI 10.1109/TrustCom.2013.248

Publicación E:

Real-time Emotional Agent Architecture Application on Service Mobile Robot Control

Carlos Domínguez, Houcine Hassan, Alfons Crespo.

The 2009 International Conference on Artificial Intelligence

Otras Publicaciones
Relacionadas con esta Tesis

“Multicore Parallel Design of Robot Control Architectures”
Carlos Domínguez, Houcine Hassan, Vicent Mayans, Alfons Crespo
ISPA’2013 – The 11th IEEE International Symposium on Parallel and Distributed Processing with Applications

“Emotional Contribution Process Implementations on Parallel Processors”
Carlos Domínguez, Houcine Hassan, José Albaladejo, Maria Marco, Alfons Crespo
ICA3PP’2011 - The 11th International Conference on Algorithms and Architectures for Parallel Processing
ISBN: 978-3-642-24668-5

“Embedded GPS Path Planning for Mobile Robots”
Luis Albert, Víctor Planes, Carlos Domínguez, Houcine Hassan
ESA’2010 - The 2010 International Conference on Embedded Systems and Applications
ISBN: 1-60132-141-4

“Simulation Framework for Validation of Emotional Agents”
Carlos Domínguez, Houcine Hassan, José Albaladejo, Alfons Crespo
ICAI’2010 - The 2010 International Conference on Artificial Intelligence
ISBN: 1-60132-148-1

“Real-Time Linux Framework for Designing Parallel Mobile Robotic Applications”
Joan Aracil, Carlos Domínguez, Houcine Hassan, and Alfons Crespo
ICA3PP’2010 - The 10th International Conference on Algorithms and Architectures for Parallel Processing
ISBN: 978-3-642-13136-3

“Simulation Tool Incorporating a Flexible Real-Time Task Model for Mobile Robotic Systems”
Jesús García, Houcine Hassan, Carlos Domínguez, Alfons Crespo
ESA’2009 - The 2009 International Conference on Embedded Systems and Applications
ISBN: 1-60132-102-3

“Flexible Real-time Embedded Architecture for Advanced Service Robots”
David Ramada, Carlos Domínguez, Houcine Hassan, Alfons Crespo
IJSH - The International Journal of Smart Home
October 2008, Volume 2, Number 4, Pages 59-74
ISSN: 1975-4094

“Real-time Embedded Architecture for Improving Pervasive Robot Services”
Carlos Domínguez, Houcine Hassan, Alfons Crespo
IJSEIA - International Journal of Software Engineering and Its Applications
January 2008, Volume 2, Number 1
ISSN: 1738-9984

“Emotional Architecture for Robotic Agents”
Carlos Domínguez, Houcine Hassan, Alfons Crespo
ICAI’2008 - The 2008 International Conference on Artificial Intelligence
ISBN: 1-60132-070-1

“Real-time Embedded Architecture for Advanced Service Robots”
David Ramada, Carlos Domínguez, Houcine Hassan, Alfons Crespo
ESA’2008 - The 2008 International Conference on Embedded Systems and Applications
ISBN: 1-60132-065-5

“Real-Time Embedded Architecture for Pervasive Robots”
Carlos Domínguez, Houcine Hassan, Alfons Crespo
IPC’2007 - The 2007 International Conference on Intelligent Pervasive Computing
ISBN: 0-7695-3006-0

“Behavioural Affective Modulation of Service Robotic Agents”
Carlos Domínguez, Houcine Hassan, Alfons Crespo
IECON'2006 - 32th Annual Conference of the IEEE Industrial Electronics Society
ISBN: 1-4244-0136-4

“Emotional Sensibility and Behavioural Control of Real-Time Robotic Agents”
Carlos Domínguez, Houcine Hassan, Alfons Crespo
ISORA’2006 - 11th International Symposium on Robotics and Applications
ISBN: 1-889-335-32-0

"Real-time Emotional Architecture for Robotic Agents"
Carlos Domínguez, Houcine Hassan, Alfons Crespo
ETFA'2005 - 10th IEEE International Conference on Emerging Technologies and Factory Automation
ISBN: 0-7803-9402-X

"Behavioural Adaptation of Real-Time Emotional Robotic Agents"
Carlos Domínguez, Houcine Hassan, Alfons Crespo
IFAC WC'2005 - 16th IFAC World Congress
ISBN: 0-08-045108-X

Financiación

Parte de esta tesis se ha desarrollado con el soporte de las siguientes entidades:

European Commission – FP7

"DREAMS - Distributed REal-time Architecture for Mixed Criticality Systems"

Generalitat Valenciana - Conselleria de Empresa Universitat i Ciència

"Modelo de Agente Emocional de Tiempo Real" GV04B/431

Universitat Politècnica de València

"Arquitectura de Alto Rendimiento basada en GPU como Soporte de Ejecución de Procesos Emocionales Altamente Paralelizables" SP20120833

"Arquitectura de Procesadores Heterogénea de Altas Prestaciones para Robots Inteligentes" 11-2906

"Diseño de Agentes Emocionales de Tiempo Real e Implementación de un Prototipo sobre Arquitecturas de Procesadores de Altas Prestaciones" PAID-05-09-4349

Bibliografía

1. Malfaz M, Salichs MA (2010) Using MUDs as an experimental platform for testing a decision making system for self-motivated autonomous agents. Artificial Intelligence and the Simulation of Behaviour Journal. 2(1): 21-44

2. Damiano L, Cañamero L (2010) Constructing Emotions. Epistemological groundings and applications in robotics for a synthetic approach to emotions. Proceedings of AI-Inspired Biology Symposium (AIIB'2010): 20-28

3. Hawes N, Wyatt J, Sloman A (2009) Exploring design space for an integrated intelligent system. Knowledge Based Systems 22(7): 509-515

4. Sloman A (2009) Some Requirements for Human-Like Robots: Why the Recent Over-Emphasis on Embodiment Has Held Up Progress. Creating Brain-Like Intelligence 2009: 248-277

5. Moravec H (2009) Rise of the Robots. The Future of Artificial Intelligence. Scientific American. 23 March 2009.

6. Moshkina L, Arkin RC (2009) Beyond Humanoid Emotions: Incorporating Traits, Attitudes and Moods. IEEE International Conference on Robotics and Automation (ICRA'2009)

7. iRobot industrial robots website. http://www.irobot.com/gi/ground/. Access 22 September 2014

8. Lee-Johnson CP, Carnegie DA (2010) Mobile Robot Navigation Modulated by Artificial Emotions. IEEE Transactions On Systems, Man, And Cybernetics - Part B. 40(2): 469-480

9. Daglarli E, Temeltas H, Yesilogly M (2009) Behavioral task processing for cognitive robots using artificial emotions. Neurocomputing 2009

10. Ventura R, Pinto-Ferreira C (2009) Responding efficiently to relevant stimuli using an emotion-based agent architecture. Neurocomputing 2009

11. Del Campo I, Basterretxea K, Echanobe J, Bosque G, Doctor F (2012) A System-on-Chip Development of a Neuro–Fuzzy Embedded Agent for Ambient-Intelligence Environments. IEEE Transactions on Systems, Man, and Cybernetics - Part B. 42(2): 501-512

12. Altera Corporation (2010) Stratix III Device Handbook vol. 1

13. Altera Corporation (2012) Stratix IV Device Handbook vol. 1

14. Intel Corporation (2014) Desktop 4th Generation Intel Core Processor Family, Desktop Intel Pentium Processor Family, and Desktop Intel Celeron Processor Family – Datasheet vol. 1 and 2

15. Pedraza C, Castillo J, Martínez JI, Huerta P, Bosque JL, Cano J (2011) Genetic Algorithm for Boolean minimization in an FPGA cluster. The Journal of Supercomputing. 58(2): 244-252

16. Orlowska-Kowalska T, Kaminski M (2011) FPGA Implementation of the Multilayer Neural Network for the Speed Estimation of the Two-Mass Drive System. IEEE Transactions on Industrial Informatics. 7(3): 436-445

17. Arkin RC, Ulam P, Wagner AR (2012) Moral Decision-making in Autonomous Systems: Enforcement, Moral Emotions, Dignity, Trust and Deception. Proceedings of the IEEE. 100(3): 571-589

18. Salichs MA, Malfaz M (2012) A new approach to modelling emotions and their use on a decision making system for artificial agent. IEEE Transactions on affective computing. In Press. 3(1): 56-68

19. Naouar MW, Monmasson E, Naassani AA, Slama-Belkhodja I, Patin N (2007) FPGA-based current controllers for AC machine drives - A review. IEEE Transactions on Industrial Electronics. 54(4): 1907–1925

20. March JL, Sahuquillo J, Hassan H, Petit S, Duato J (2011) A New Energy-Aware Dynamic Task Set Partitioning Algorithm for Soft and Hard Embedded Real-Time Systems. The Computer Journal. 54(8):1282-1294

21. Lehoczky J, Sha, L, Ding Y (1989) The rate monotonic scheduling algorithm: exact characterization and average case behavior. IEEE Real-Time Systems Symposium, Santa Monica, Dec. 1989: 166–171

22. Seo E, Jeong J, Park S, Lee J (2008) Energy efficient scheduling of real-time tasks on multicore processors. IEEE Transactions on Parallel and Distributed Systems. 19(11): 1540–1552.

23. Ng-Thow-Hing V, Lim J, Wormer J, Sarvadevabhatla RK, Rocha C, Sakagami Y (2008) The memory game: Creating a human robot interactive scenario for ASIMO. IEEE/RSJ

International Conference on Intelligent Robots and Systems (IROS'2008): 779–786.

24. Thu Bui L, Abbass HA, Barlow M, Bender A (2012) Robustness Against the Decision-Maker's Attitude to Risk in Problems With Conflicting Objectives. IEEE Transactions on Evolutionary Computation. 16(1): 1-19

25. Pedrycz W, Song M (2011) Analytic Hierarchy Process (AHP) in Group Decision Making and its Optimization with an Allocation of Information Granularity. IEEE Transactions on Fuzzy Systems. 19(3): 527-539

26. Cassidy AS, Merolla P, Arthur JV, Esser SK, Jackson B, Alvarez-icaza R, Datta P, Sawada J, Wong TM, Feldman V, Amir A, Ben-dayan D, Mcquinn E, Risk WP, Modha DS (2013) Cognitive computing building block: A versatile and efficient digital neuron model for neurosynaptic cores. IEEE International Joint Conference on Neural Networks (IJCNN'2013).

27. IBM Cognitive Computing and Neurosynaptic chips website. http://www.research.ibm.com/cognitive-computing/neurosynaptic-chips.shtml. Access 22 September 2014

28. C. Domínguez, H. Hassan, A. Crespo (2009), Real-time Emotional Agent Architecture Application on Service Mobile Robot Control The 2009 International Conference on Artificial Intelligence. Las Vegas, Nevada, USA.

29. C. Dominguez, H. Hassan, A. Crespo and J. Albaladejo (2014), Multicore and FPGA implementations of emotional-based agent architectures, The Journal of Supercomputing, DOI: 10.1007/s11227-014-1307-6 (published Online October 2014).

30. M.R. Jamali, A. Arami, M. Dehyadegari, C. Lucas, Z. Navabi (2009), Emotion on FPGA: Model driven approach, Expert Systems with Applications 36 (2009) 7369?7378.

31. A. Ducrot, Y. Dumortier, I. Herlin, V. Ducrot (2011), Real-time quasi dense two-frames depth map for Autonomous Guided Vehicles, Intelligent Vehicles Symposium (IV), 2011 IEEE, vol., no., pp.497-503, 5-9 June 2011

32. J.T. Kider, M. Henderson, M. Likhachev, A. Safonova (2010), Highdimensional planning on the GPU, 2010 IEEE International Conference on Robotics and Automation (ICRA), pp.2515-2522, 3-7 May 2010.

33. Cuda (2010), NVIDIA CUDA C Programming Guide, NVIDIA, Nov 2010.

34. M. Harris (2012), Nvidia developer zone, http://devblogs.nvidia.com/parallelforall/how-overlap-data-transfers-cudacc/

35. J. Luitjens, S. Rennich (2011), CUDA Warps and Occupancy, http://on-demand.gputechconf.com/gtcexpress/2011/presentations/cuda webinars WarpsAndOccupancy.pdf

36. J. Stokes (2000), SIMD architectures, http://arstechnica.com/features/2000/03/simd/2/

37. C. Lomont, Introduction to Intel Advanced Vector Extensions, http://software.intel.com/en-us/articles/introduction-to-intel-advancedvector-extensions

38. M. Shigeo (2010), Fast approximate oat function Fmath, http://homepage1.nifty.com/herumi/soft/fmath.html

39. Intel Core i7-870 Processor Specifications, http://ark.intel.com/products/41315/IntelCorei7870Processor8M Cache2 93-GHz

40. Intel Processor Core i7 2600, http://ark.intel.com/products/52213.

41. Brandenburg, B.B.; Calandrino, J.M.; Anderson, J.H., "On the Scalability of Real-Time Scheduling Algorithms on Multicore Platforms: A Case Study," Real-Time Systems Symposium, 2008 , vol., no., pp.157,169, Nov. 30 2008-Dec. 3 2008.

42. C. Domínguez, H. Hassan, A. Crespo (2006) “Behavioural Affective Modulation of Service Robotic Agents” 32th Annual Conference of the IEEE Industrial Electronics Society.

43. C. Domínguez, H. Hassan, A. Crespo (2008) “Real-time Embedded Architecture for Improving Pervasive Robot Services” International Journal of Software Engineering and Its Applications, January 2008, Volume 2, Number 1.

44. O. Pereira-Zapata, P. Mejia-Alvarez (2003). Analysis of Real-Time Multiprocessors Scheduling Algorithms. IEEE Real Time Systems Symposium, 2003.

45. Limsoonthrakul, S.; Dailey, M.N.; Srisupundit, M.; Tongphu, S.; Parnichkun, M., "A modular system architecture for autonomous robots based on blackboard and publish-subscribe

mechanisms," IEEE International Conference on Robotics and Biomimetics, 2008. ROBIO 2008., vol., no., pp.633,638, 22-25 Feb. 2009

46. Arkin, R.C., Ali, K., Weitzenfeld, A., and Cervantes-Perez, F., (2000). Behavioral Models of the Praying Mantis as a Basis for Robotic Behavior, Journal of Robotics and Autonomous System, 32 (1), 39-60.

47. Ronald C. Arkin, Fujita M., Takagi T. and Hasegawa R. (2003) "An Ethological and Emotional Basis for Human-Robot Interaction", Journal of Robotics and Autonomous Systems, 42 (2003) 191-201.

48. Moshkina, L., and Arkin, R.C., 2003, "On TAMEing Robots", Proc. IEEE International Conference on Systems, Man and Cybernetics, Oct. 2003.

49. Breazeal C., 2003, "Emotion and Sociable Humanoid Robots". E. Hudlika (ed), International Journal of Human Computer Studies, 59, pp. 119-155.

50. Arbib M, Fellous J. M., (2004). "Emotions: from Brain to Robots", TRENDS in Cognitive Sciences Vol.8 No.12 December 2004.

51. Sloman, A., Chrisley, R., Scheutz, M., (2004). The Architectural Basis of Affective States and Processes. In: Who Needs Emotions? The Brain Meets the Robot. (JM Fellous and M. A. Arbib. Ed.). Oxford University Press.

52. Nair, R., Tambe, M., and Marsella S., (2004). The Role of Emotions in Multiagent Teamwork. In: Who Needs Emotions? The Brain Meets the Robot. (JM Fellous and M. A. Arbib. Ed.). Oxford University Press.

53. Hudlicka, E. (2004). Two Sides of Appraisal: Implementing Appraisal and Its Consequences within a. Cognitive Architecture. In Proceedings of the Sixth International Conference on Cognitive Modeling, 118-123.

54. N.C. Audsley, A. Burns, R. Davis, K. Tindell and J. Wellings "Fixed Priority Pre-emptive Scheduling: An Historical Perspective". The Journal of Real-Time Systems, Vol. 8. pp. 173-198. 1995.

55. M. Barabanov and V. Yodaiken, "Introducing real-time Linux". Linux Journal 34, Feb 1997.

56. M.A.Salichs and M.Malfaz. Using Emotions on Autonomous Agents. The Role of Happiness, Sadness and Fear. In Proceedings of AISB'06: Adaptation in Artificial and Biological Systems. University of Bristol, Bristol, England. April, 2006.