Universidad de Concepción

Introduction to Multi-Agent Systems

Dr. John Atkinson

atkinson@inf.udec.cl

Contents

- Motivation
- Agents
- Multi-Agent Systems
- Interaction and Communication
- Conclusions

Motivation

- Recent trends in history of computing:
  - *Ubiquity*
  - *Interconnection*
  - *Delegation*
  - *Intelligence*
  - *Human oriented*

Motivation

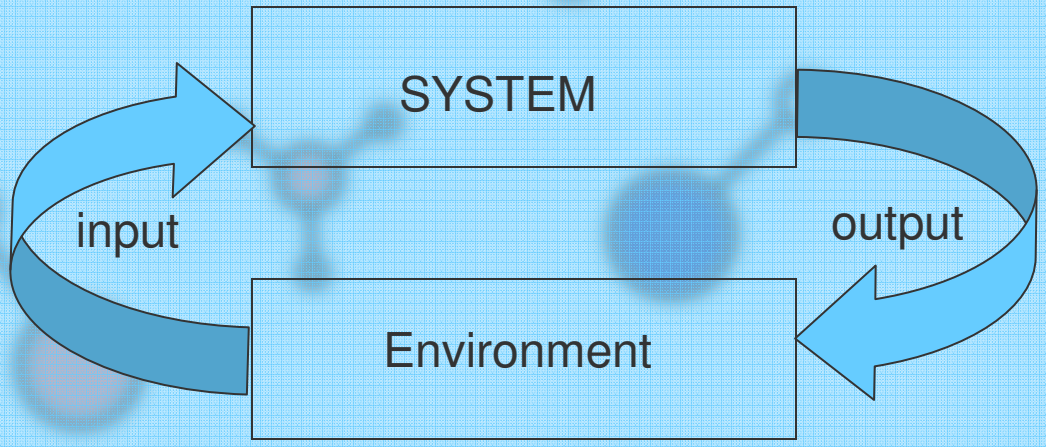
- *Los sistemas actuales tienen tres desafíos principales*:
  - Los sistemas deben ser capaces de actuar *independientemente*
  - Los sistemas deben actuar en forma tal que *represente nuestros mejores intereses* mientras interactua con otros humanos o sistemas.
  - Los sistemas (varias entidades o “agentes”) deben interconectarse para resolver problemas complejos.

An Agent: Informally

- *Un agente puede verse como un sistema computacional que es capaz de actuar independientemente en nombre de un usuario.*
- Para actuar, un agente debe ser capaz de percibir su ambiente a través de sensores y realizar acciones sobre dicho ambiente a través de efectores.
- *La independencia implica la habilidad de un agente para entender QUE es lo que se debe realizar para satisfacer ciertos objetivos y no que se le diga COMO llevarlo a cabo.*

¿What is an Agent?

- Agentes son sistemas computacionales autónomos: capaces de actuar en forma independiente mostrando control sobre sus estados internos.
- Luego, un agente es capaz de actuar en forma flexible en cierto ambiente con el fin de satisfacer sus objetivos de diseño.



¿What is an Agent?

- Flexibility means that an agent must be:
  - *Reactive*
  - *Proactive*
  - *Social*

An Agent and its Environment

- Asuma que un ambiente se puede encontrar en cualquiera de un conjunto finito E *de estados instantáneos discretos:*

$$E=\{e,e',\ldots\}$$

- Se asume que un agente tiene un repertorio de posibles acciones disponibles, que transforman el estado del ambiente:

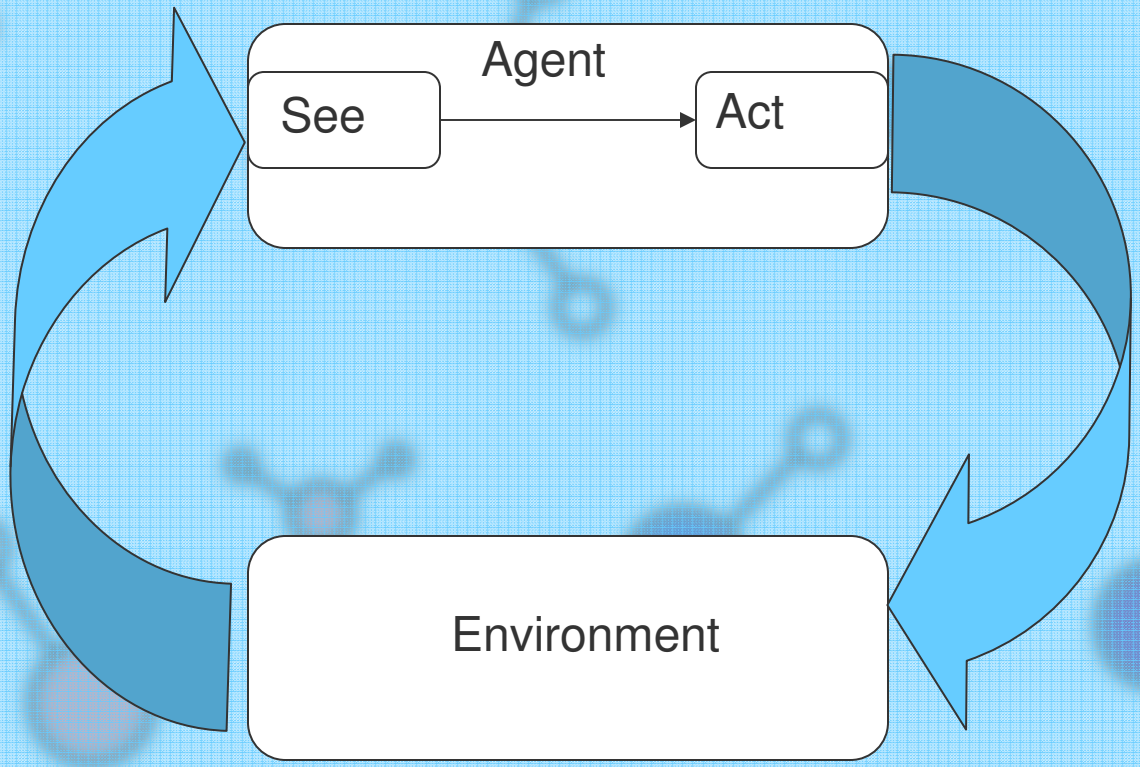
$$Ac=\{\alpha, \alpha',..\}$$

- Una ejecución, r, de un agente sobre un ambiente es una secuencia de estados y acciones:

$$r: \; e_0 \xrightarrow{\alpha_0} e_1 \xrightarrow{\alpha_1} e_2 \xrightarrow{\alpha_2} e_3 \xrightarrow{\alpha_3} e_4 \xrightarrow{\alpha_4} \ldots$$

Agents and its Perception

Input for an agent comes from its *perception* system:



Control for an Agent

1. Agente comienza en algún estado interno inicial i_0
2. Observa su estado del ambiente e, y genera una percepción
3. El estado interno del agente se actualiza para luego producir el siguiente estado
4. La acción seleccionada por el agente es la dada por el nuevo estado
5. Ir a paso 2

Designing Agents

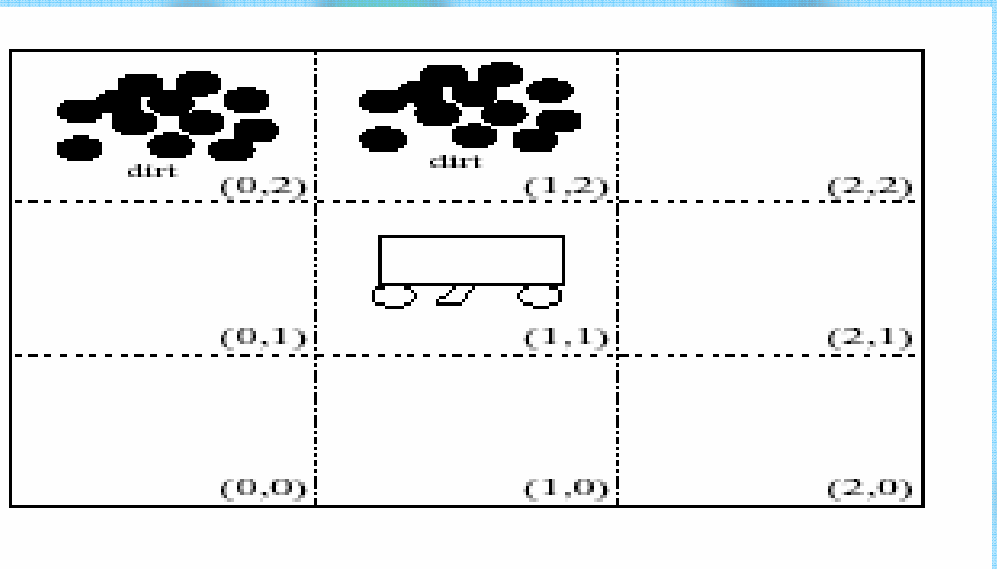
- Existen tres paradigmas básicos para concebir sistemas basados en agentes:
  - Symbolic (deliberative)
  - Reactive
  - Hybrid
- Cada paradigma lleva asociado diferentes enfoques arquitectónicos para diseñar agentes.

Symbolic Reasoning Agents

- Los agentes se ven como un tipo particular de sistema basado en conocimiento y razonamiento (deliberativo).
- Este paradigma se conoce como Inteligencia Artificial Simbólica.
- Definimos un agente deliberativo como un sistema que:
  - Contiene un modelo simbólico del mundo explícitamente representado.
  - Toma decisiones (ej. acerca de que acción realizar) a través del razonamiento simbólico.

Reasoning Agents

- An example: the “vacuum cleaner” world
  - Un agente debe limpiar toda la basura



Reasoning Agents

- Algunas reglas “lógicas” para decidir que hacer:

$$
\begin{aligned}
In(0,0) \wedge Facing(north) \wedge \neg Dirt(0,0) &\longrightarrow Do(forward) \\
In(0,1) \wedge Facing(north) \wedge \neg Dirt(0,1) &\longrightarrow Do(forward) \\
In(0,2) \wedge Facing(north) \wedge \neg Dirt(0,2) &\longrightarrow Do(turn) \\
In(0,2) \wedge Facing(east) &\longrightarrow Do(forward)
\end{aligned}
$$

- Utilizando estas reglas y comenzando en $(0, 0)$, el agente limpiará la basura.

Reasoning Agents

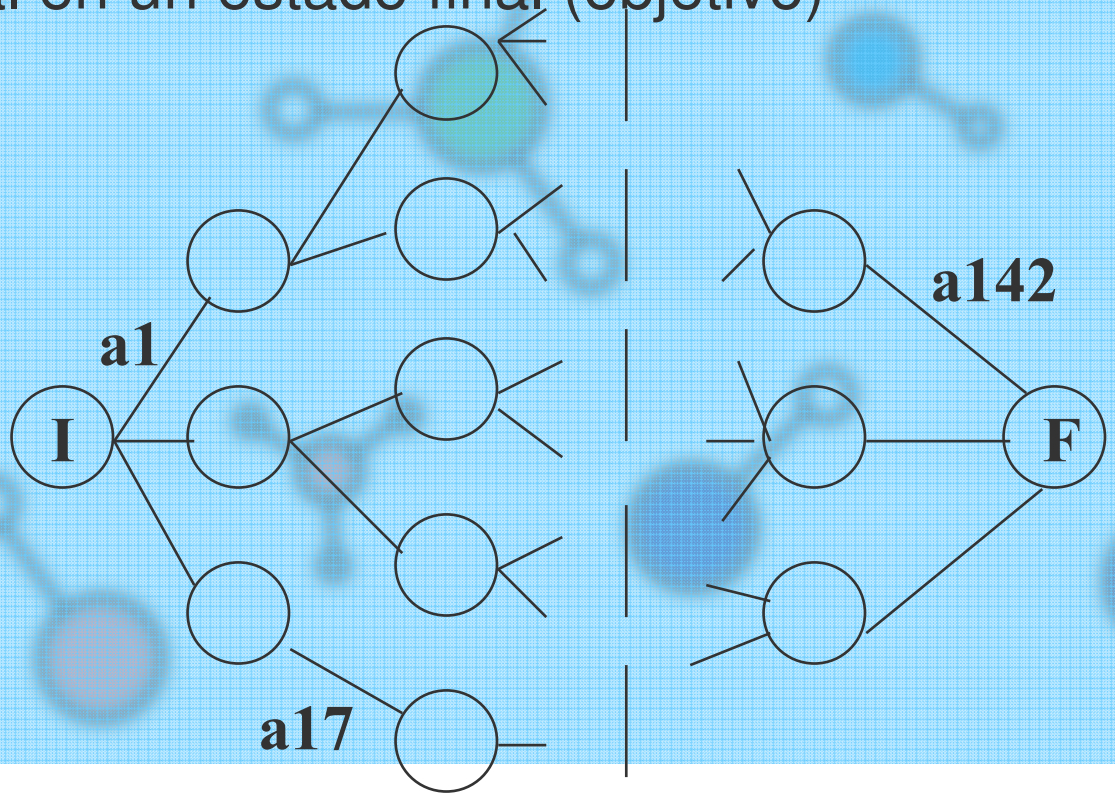
- Problemas acerca de la toma de decisiones:
  - Se asume un ambiente estático
  - La utilización de lógica de primer orden es indecidible!
- Incluso si utilizamos lógica proposicional, la toma de decisiones implica resolver problemas NP-completos.
- Soluciones típicas: “relajar” la lógica, utilizar representaciones simbólicas no lógicas.

More problems…

- El enfoque lógico involucra agregar y eliminar cosas de una base de datos.
- Eso NO es lógica pura
- *Alternativa*: crear agentes de planificación que utilicen deducción lógica para resolver el problema.

Planning Systems

- Los sistemas de planificación encuentran una secuencia de acciones que transforma un estado inicial en un estado final (objetivo)



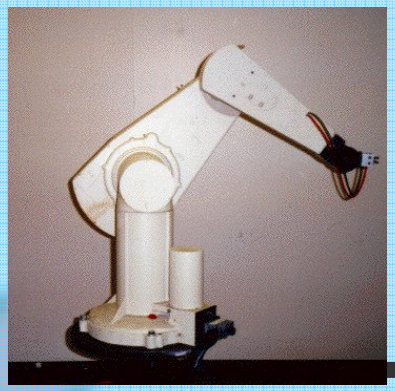
Planning for Agents

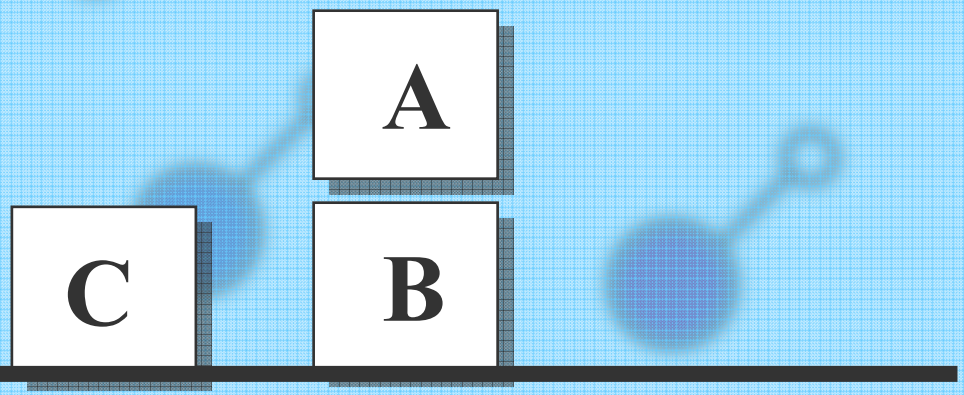
- Planificación involucra aspectos relacionados con la *Búsqueda* y la *Representación de Conocimientos*.
- Algunos enfoques de planificación:
  - Planificación de Robots (STRIPS)
  - Planificación de experimentos biológicos (MOLGEN)
  - Planificación de Actos de Habla (Speech Acts)

Example: the Blocks World

- Utilizamos predicados lógicos para describir el mundo:

```
ON(A,B)          ARMEMPTY
ONTABLE(B)       ONTABLE(C)    …
```





Describing Facts

- Podemos escribir verdades lógicas generales relacionadas a los predicados:

$$[\ \exists\ x\ \mathrm{HOLDING}(x)\ ] \rightarrow \neg\ \mathrm{ARMEMPTY}$$

$$\forall\ x\ [\ \mathrm{ONTABLE}(x) \rightarrow \neg\ \exists\ y\ [\mathrm{ON}(x,y)]\ ]$$

$$\forall\ x\ [\ \neg\ \exists\ y\ [\mathrm{ON}(y, x)] \rightarrow \mathrm{CLEAR}(x)\ ]$$

Problems

¿Cómo determinamos lo que cambia y no cambia en el mundo cuando se realiza una acción?

- Especificar los predicados que permanecen sin cambios después de una acción (*Frame Axioms*).

Ejemplo:

ACCION:	go(Agent,From,To)
PRECONDICIONES:	at(Agent,From)
LISTA ADD:	at(Agent,To)
LISTA DELETE:	at(Agent,From)

Sin embargo…

- Este método nos fuerza a escribir muchos *frame axioms*
- *N*ecesitamos otras alternativas de razonamiento para el agente!!.

Solución:

- Razonar sobre los “estados mentales” (intenciones) de un agente!!

Intentional Systems

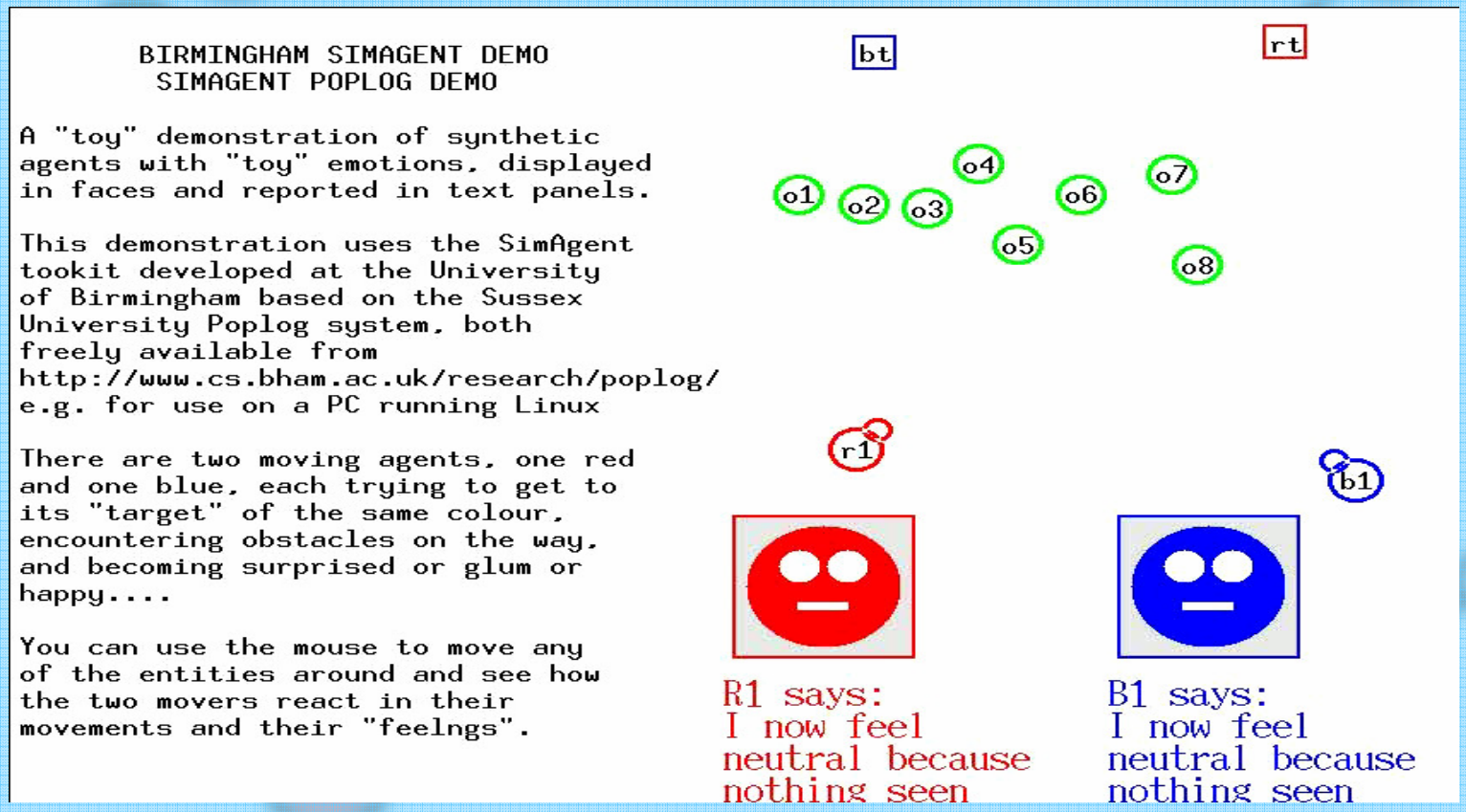
- Como seres humanos utilizamos aspectos intencionales como un mecanismo de *abstracción para representar las propiedades de sistemas complejos.*

- *La postura intencional es una estrategia que permite interpretar la conducta de una entidad tratándola como si esta fuera un agente racional que controla su "elección" de "acción", considerando sus "creencias" y "deseos"*

Intentional Systems

- Idea básica:
  - "programar" los agentes en términos de nociones intencionales tales como *creencias, compromisos e intenciones*.
- Los agentes intencionales se fundamentan en la teoría de intenciones y el modelo BDI (*Belief-Desire-Intention*)
- Esto origina un nuevo paradigma de programación denominado *Agent-Oriented Programming (AOP)*.

Intentional Agents



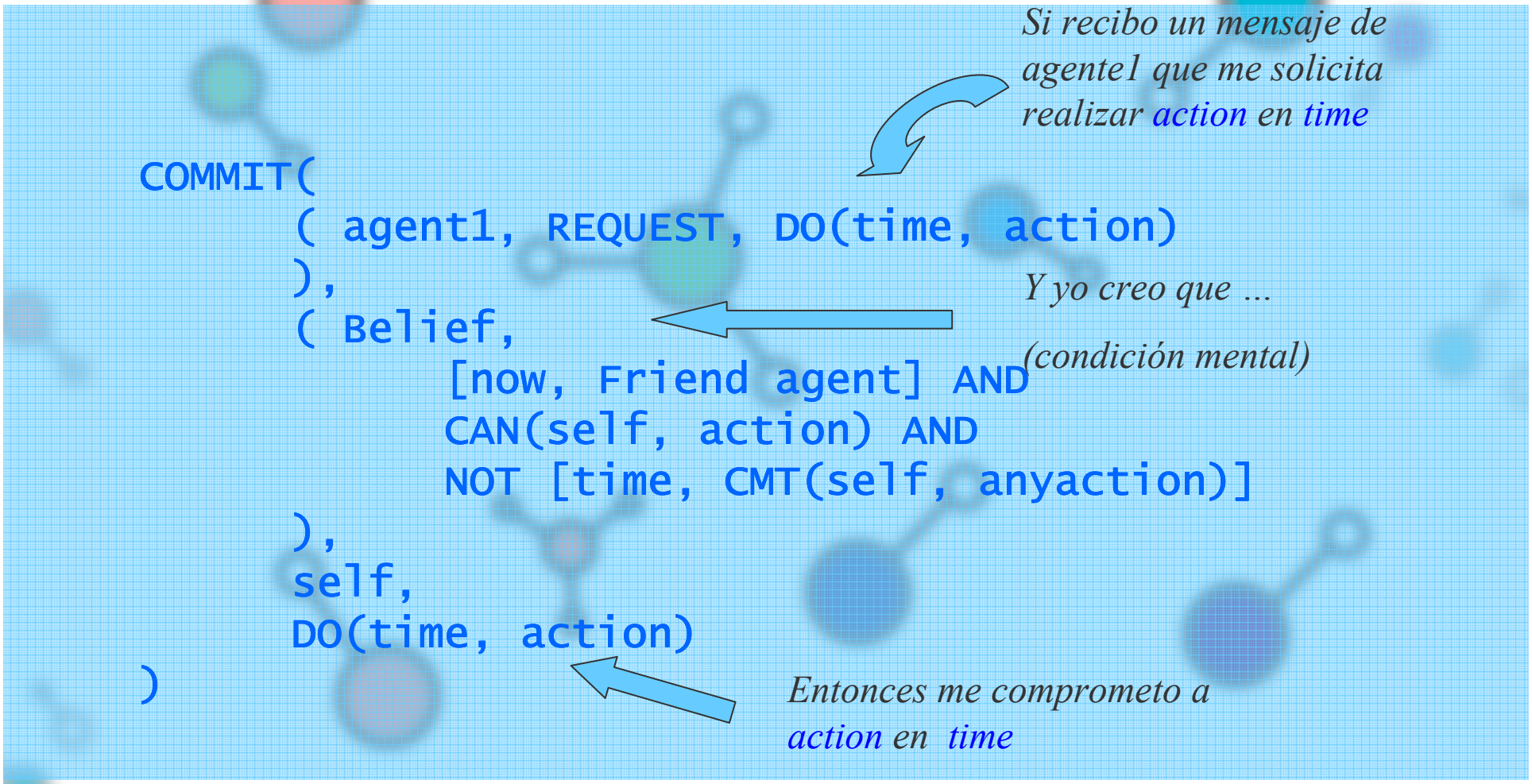
AOP

- AGENT0 fue el primer lenguaje de programación orientado a un sistema de agentes.
- Cada agente en AGENT0 tiene capacidades, creencias y compromisos iniciales y reglas de compromiso.
- Los agentes se comunican a través de mensajes.
- AGENT0 se basa en la teoría de *Speech Acts* (Austin), los mensajes estan restringidos a uno de tres tipos:
  - "request" para comprometerse a una acción
  - "unrequest" para no realizar una acción
  - "inform" para traspasar información

AGENT0

- En cada "*ciclo del agente*":
  - La condición del mensaje se compara con los mensajes que ha recibido el agente
  - La condición mental se compara con las creencias del agente
  - Si la regla se satisface, el agente se compromete a la acción (la acción se agrega al conjunto de compromisos del agente)

A "Commitment"



Practical Reasoning

- Para que los agentes operen con intenciones, se requiere "razonar" sobre las acciones posibles.
- Razonamiento práctico humano consiste de dos actividades:
  - *deliberación*
    decidir qué estado deseamos alcanzar
  - *Razonamiento mean-ends*
    decidir cómo alcanzar estos estados
- Las salidas de la deliberación son *intenciones*

Mean-Ends Reasoning

- El razonamiento de medios-fines, proporciona a un agente:
  - representación de objetivo/intención a lograr
  - representación de acciones que puede realizar
  - representación del ambiente

que le permite generar un *plan* para lograr el objetivo.

Existen varios tipos de planificadores siendo *STRIPS* y *GraphPlan* los más conocidos

A Reasoning Agent

```
Agent Control
1. B = B0   /* initial beliefs */
2. while true
3.      Get next perception p;
4.      B = brf(B,p);
5.      I = deliberate(B);
6.      π=plan(B,I);
7.      execute(π);
8.  end while
```

Deliberation

- ¿Cómo delibera un agente?
  - Comienza tratando de entender cuáles son las *opciones* disponibles
  - *Elige entre ellas,* y se compromete (*commit*) *a alguna*
- Las intenciones corresponderán a las opciones elegidas.

Deliberation

- La función de *deliberación* puede descomponerse en dos elementos:
  - *Generación de opciones*
    el agente genera un conjunto de alternativas posibles;
    recibe las creencias actuales del agente y determina un conjunto de opciones (= *deseos*)
  - *Filtrado*
    el agente elige entre altenativas que compiten, y se compromete a lograrlas.

Implementing Deliberation

```
Agent Control
1. B = B0
2. I = I0
3. while true
4.      get next perception p;
5.      B = brf(B,p);
6.      D = options(B,I);
7.      I = filter(B,D,I);
8.      π=plan(B,I);
9.      execute(π);
10. end while
```

Deliberation

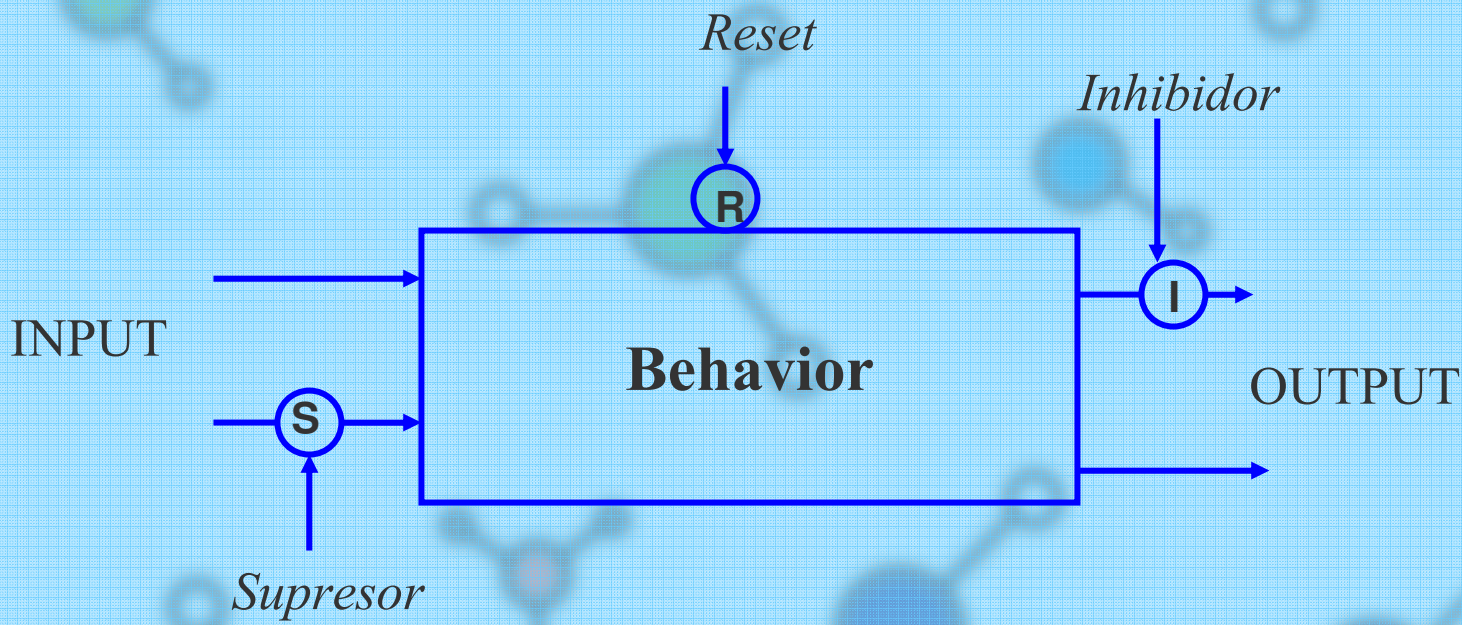
Reactive Architectures

- Este tipo de paradigma explota propiedades de un enfoque moderno de IA denominado *Inteligencia basada en el Comportamiento* (Rodney Brooks, MIT).
- Brooks estable 3 hipótesis:
  1. Se puede generar comportamiento inteligente SIN modelos explícitos del mundo.
  2. El comportamiento inteligente puede generararse SIN razonamiento abstracto explícito.
  3. La inteligencia es una propiedad *emergente* de ciertos sistemas complejos

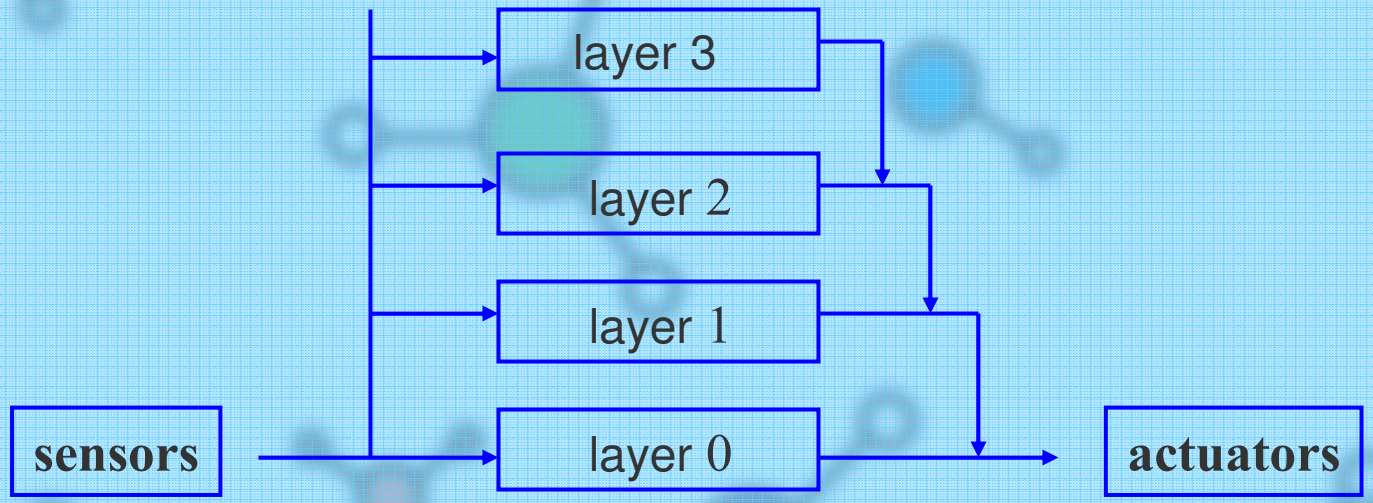
Behavior-based Intelligence

- Para ilustrar sus ideas, Brooks diseño agentes robóticos basados en una organización de subsumición (Subsumption).
- Esta arquitectura corresponde a una jerarquía de conductas que llevan a cabo tareas.
- Cada conducta “compite” con las otras para ejercer control sobre el agente
- Los niveles inferiores representan clases más primitivas de conductas (ej. `evitar obstáculos`), y tienen precedencia sobre los niveles superiores de la jerarquía

A Behavior

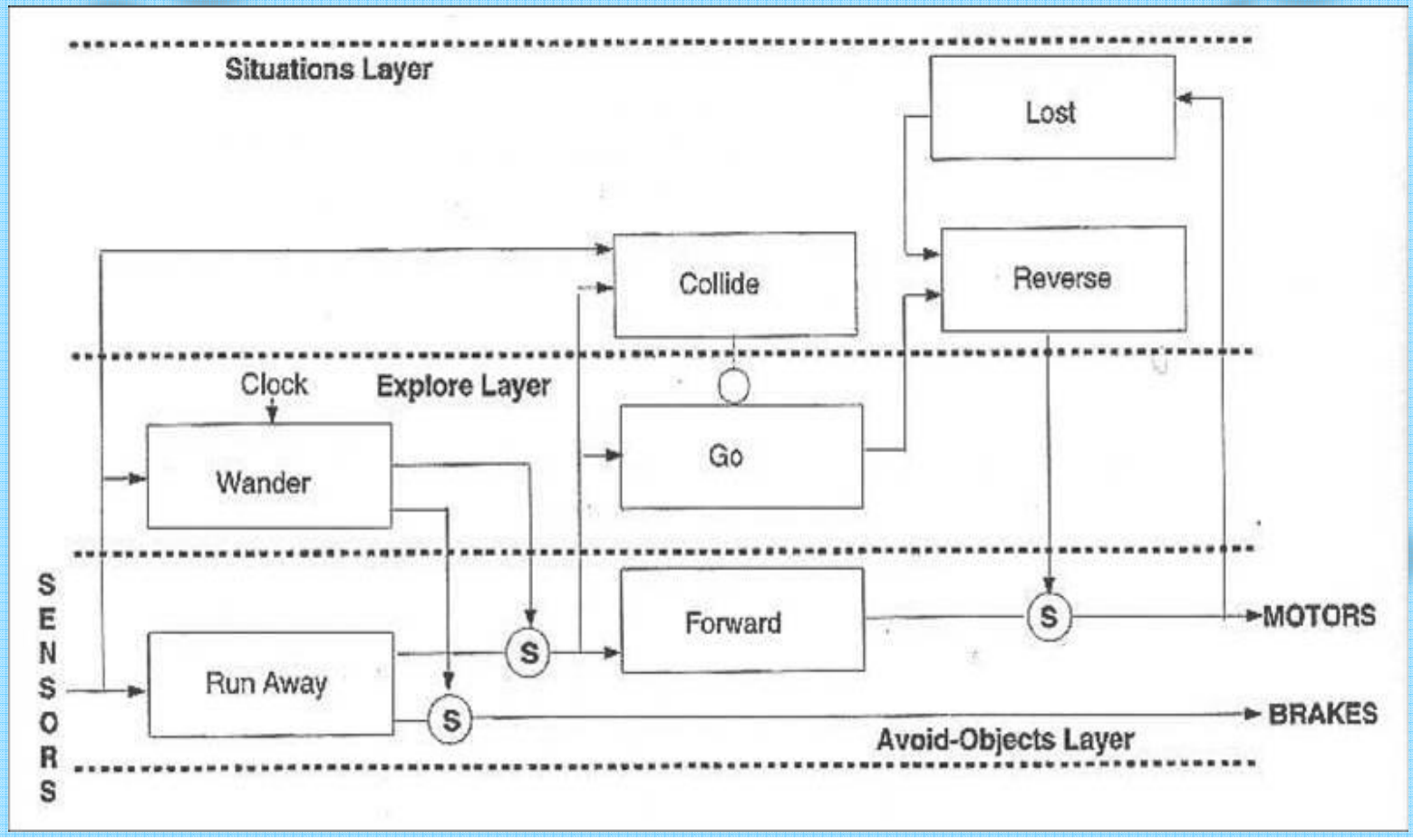


Layers and Behaviors



Emergent Behavior

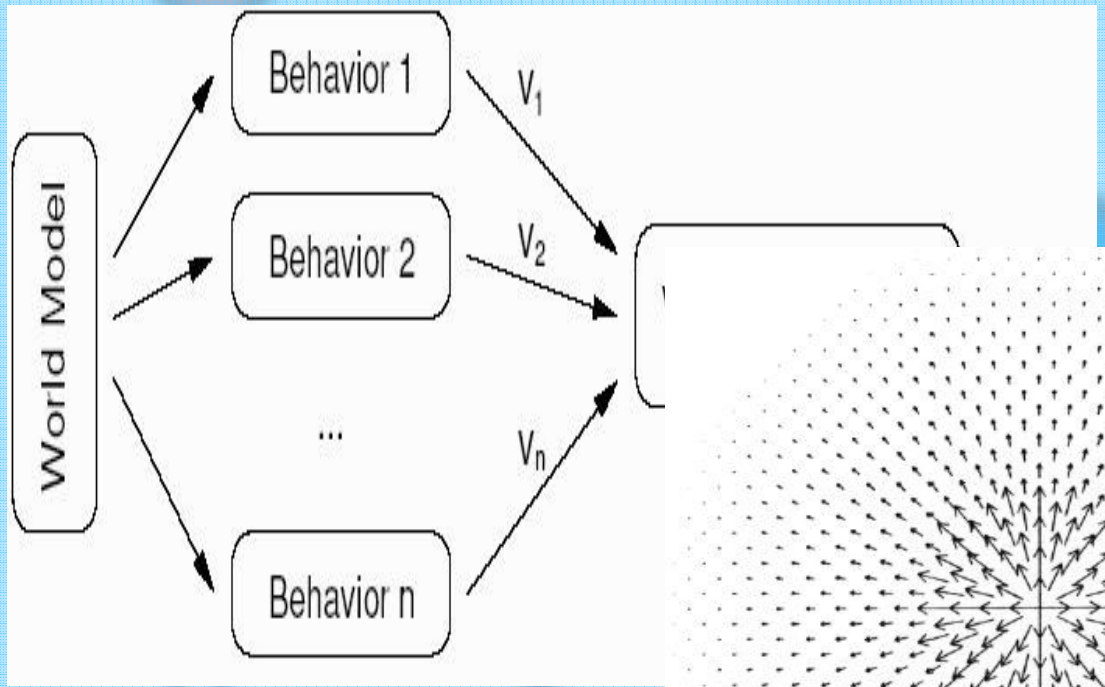
An Example

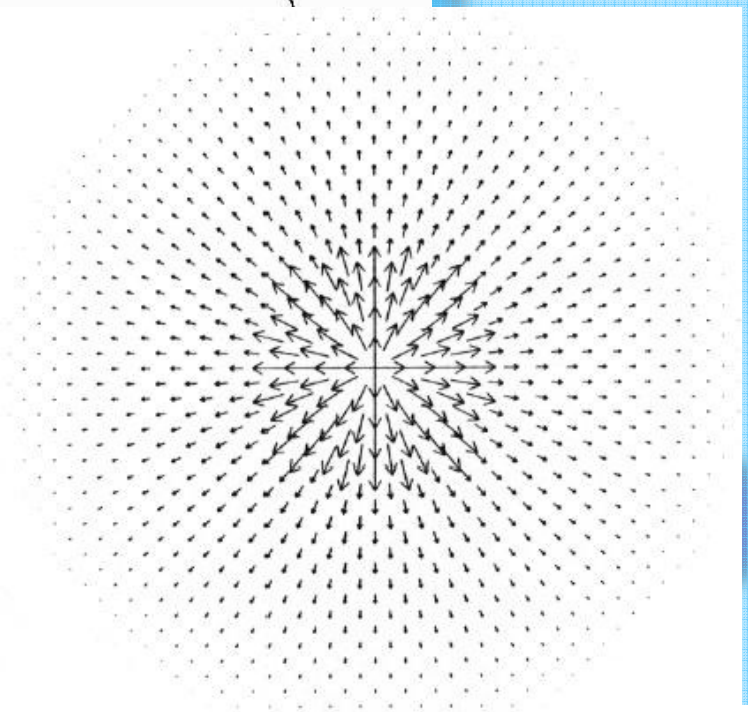

Situations Layer
Lost
Collide
Reverse
Clock
Explore Layer
Wander
Go
SENSORS
Forward
S
MOTORS
Run Away
S
S
BRAKES
Avoid-Objects Layer

Motor Schema

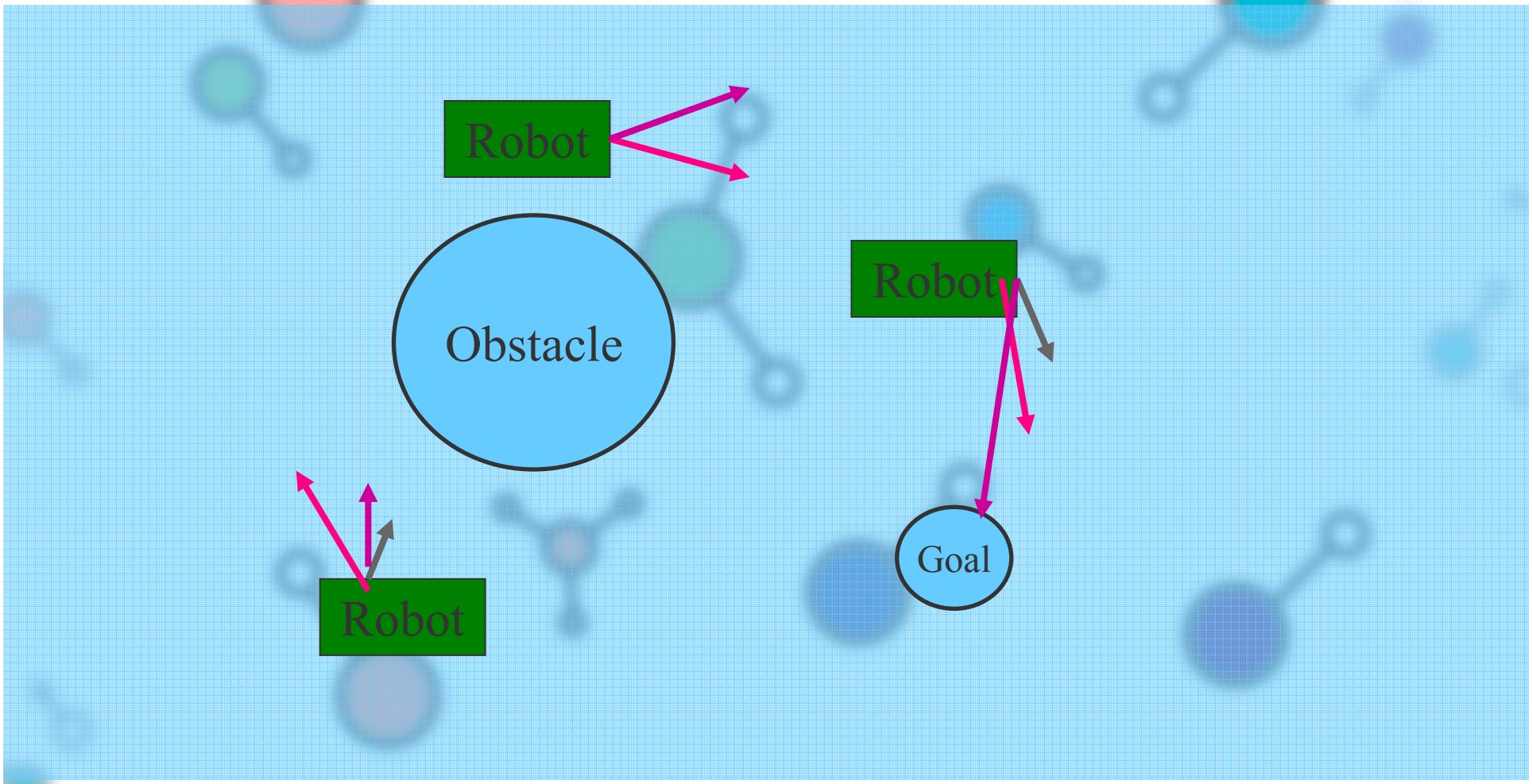
- Un tipo de arquitectura reactiva “cooperativa” para agentes alternativa se denomina *Esquemas Motores* (Arkin).
- Los esquemas motores representan las conductas como *actividades cerebrales*
- El resultado de una tarea es la “cooperación” concurrente de varios esquemas.
- Este resultado se expresa especificando las conductas como *Campos Potenciales de Vectores*.
- Las respuestas de las conductas se generan en términos de vectores.
- Las fuerzas relativas de las conductas determinan la respuesta total del agente

Potential Field based Behaviors





Example: Robot Navigation



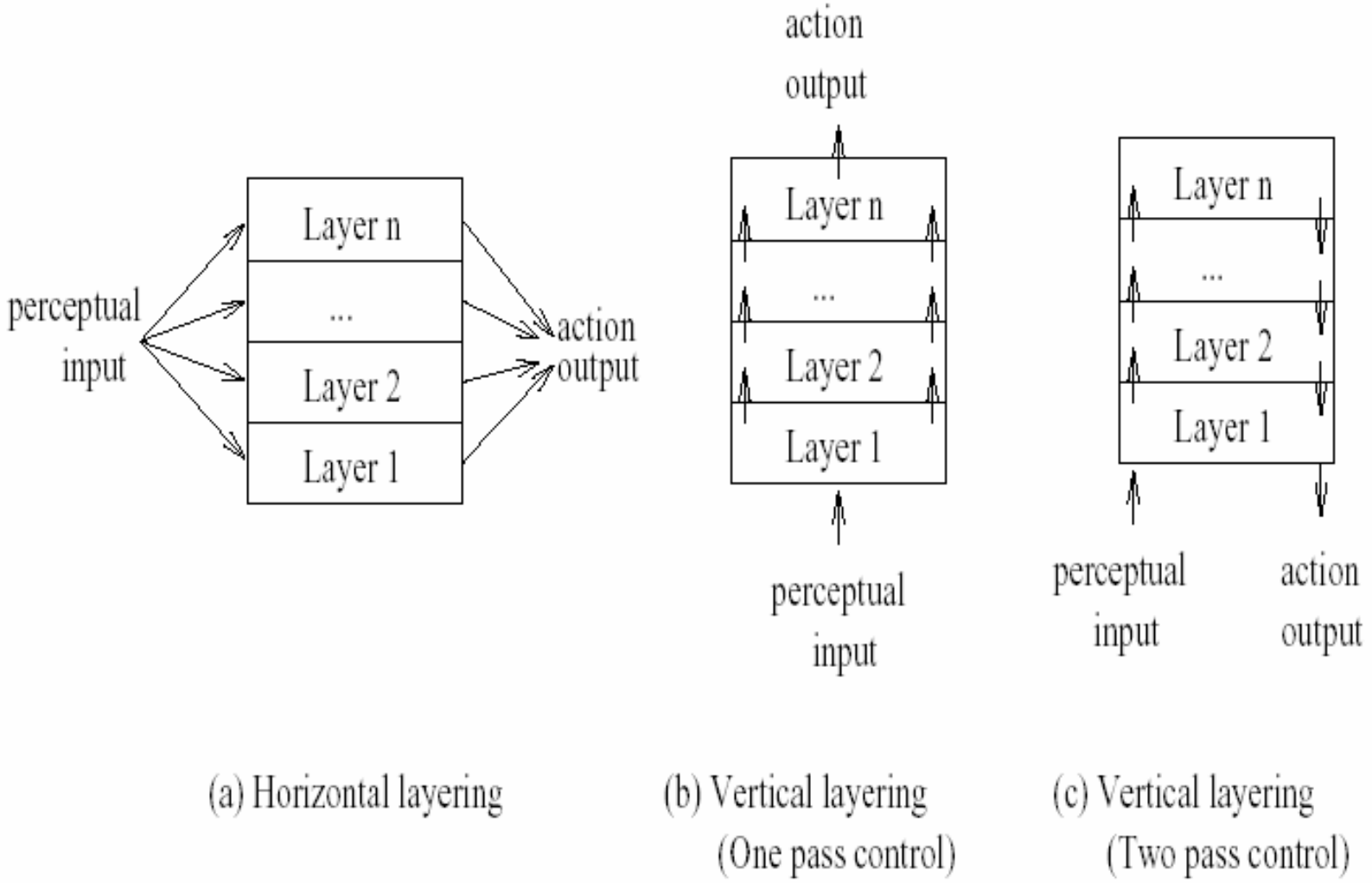
Hybrid Architectures

- La alternativa es disponer de enfoques híbridos que combinen los paradigmas clásicos con los modernos.
- Un enfoque obvio es construir un agente de dos o más subsistemas:
  - Uno deliberativo, que contenga el modelo simbólico del mundo, que desarrolla planes y que toma decisiones a la manera tradicional de la IA.
  - Uno reactivo, que sea capaz de reaccionar a eventos sin razonamiento complejo.

Hybrid Architectures

- *Horizontal layering*
  los layers estan directamente conectados a la entrada sensorial y la salida de acciones. Cada layer actua como un agente, produciendo sugerencias sobre que acción realizar.
- *Vertical layering*
  la entrada sensorial y la salida de acciones son manejadas por al menos un layer

Hybrid Architectures



(a) Horizontal layering

(b) Vertical layering
(One pass control)

(c) Vertical layering
(Two pass control)

Multi-Agent Systems

- Hasta el momento, tenemos la noción de agentes autónomos *individuales*.
- Sin embargo, en ambientes reales y dinámicos los agentes deben interactuar con otros agentes para resolver problemas comunes.
- Un *Sistema Multiagente* (SMA) contiene cierto número de agentes que …
  - Interactuan a través de la comunicación
  - Son capaces de actuar en el ambiente
  - Tienen diferentes “esferas” de influencia
  - Estan conectados por relaciones en la organización

Utilities and Preferences

- Suponga que tenemos dos agentes: $Ag = \{i, j\}$
- Y que estos son auto-motivados: tienen sus propias preferencias acerca de cómo está el ambiente.
- Sea $\Omega = \{\omega_1, \omega_2, \ldots\}$ el conjunto de acciones de "salida" sobre las cuales los agentes tienen preferencias
- Las preferencias las capturamos mediante *funciones* de *utilidad* :

$$u_i = \Omega \rightarrow \square$$
$$u_j = \Omega \rightarrow \square$$

- Estas funciones originan ordenamientos de preferencias sobre las salidas.

$\omega < \omega'$ significa que $u_i(\omega) < u_i(\omega')$
$\omega > \omega'$ significa que $u_i(\omega) > u_i(\omega')$

“Encounters” between Agents

- Necesitamos un modelo del ambiente en el cual:
  - Los agentes eligen simultáneamente una acción a realizar, y como resultado de las acciones, seleccionan una salida en Ω
  - La salida actual depende de la combinación de las acciones.
  - Asumamos que cada agente puede realizar dos posibles acciones, *C* (“cooperar”) and *D* (“desertar”)
- La conducta del ambiente dada por la función de transformación de estados es:

$$\tau : \underbrace{Ac}_{\text{agent } i\text{'s action}} \times \underbrace{Ac}_{\text{agent } j\text{'s action}} \rightarrow \Omega$$

Rational Action

- Suponga que sabemos que ambos agentes pueden influenciar la salida:

$$\begin{array}{llll} u_i(\omega_1)=1 & u_i(\omega_2)=1 & u_i(\omega_3)=4 & u_i(\omega_4)=4 \\ u_j(\omega_1)=1 & u_j(\omega_2)=4 & u_j(\omega_3)=1 & u_j(\omega_4)=4 \end{array}$$

- Lo anterior se podría ver también como:

$$\begin{array}{llll} u_i(D,D)=1 & u_i(D,C)=1 & u_i(C,D)=4 & u_i(C,C)=4 \\ u_j(D,D)=1 & u_j(D,C)=4 & u_j(C,D)=1 & u_j(C,C)=4 \end{array}$$

- Luego, las preferencias del agente i son:

$$C,C \succeq_i C,D \quad \succ_i \quad D,C \succeq_i D,D$$

- Luego, "*C*" es la elección racional para i.
  (i prefiere todas las salidas que surgen desde C sobre las salidas que surgen desde *D*)

Game Theory

- Dada una estrategia (*C* ó *D*) del agente i, habrá un número de acciones de salida posibles.
- Decimos que s_1 *domina* s_2 si toda salida posible de i jugando s_1 se prefiere sobre toda salida de i jugando s_2
- Lo anterior es una condición que debería cumplir cualquier estrategia de un agente en un MAS y se denomina *Eficiencia de Pareto.*
- Un agente racional NUNCA jugará una estrategia dominada.
- Problema: Generalmente existe más de una estrategia no dominada!!.

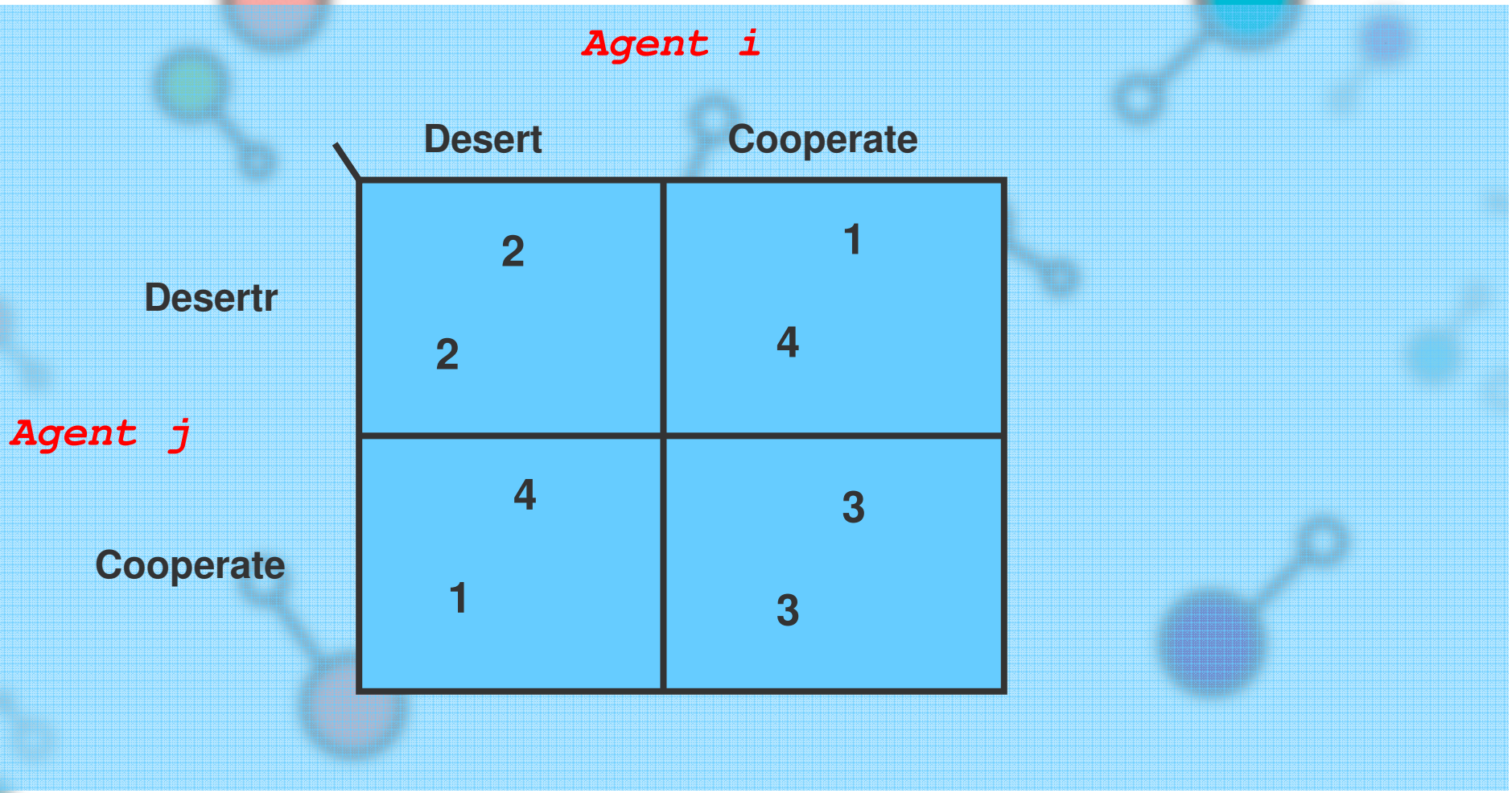
Nash Equilibrium

- Muchas estrategias que veremos provienen de la *teoría matemática de juegos (John Nash)*.
- En general, se dice que dos estrategias s_1 *y* s_2 estan en *Equilibrio de Nash* si:
  1. Suponiendo que agente i *juega* s_1, lo mejor que puede hacer el agente j *es jugar* s_2
  2. Suponiendo que agente j *juega* s_2, lo mejor que puede hacer el agente i *es jugar* s_1.
- *Ninguno de los agentes tiene un incentivo para desviarse del equilibrio de Nash.*
- Desafortunadamente, en el mundo real:
  1. *No todos los escenarios de interacción tienen un equilibrio de Nash*
  2. *Algunos escenarios tienen más de un equilibrio.*

The Prisioner Dilemma

- Dos hombres son detenidos por un crimen y colocados en celdas separadas, sin forma de reunirse o comunicarse. Se les ha dicho que:
  - Si uno confiesa y el otro no, el confesor será liberado, y el otro será encarcelado por 3 años.
  - Si ambos confiesan, cada uno será encarcelado por 2 años
- Ambos prisioneros saben que si ninguno confiesa (desertan), cada uno de ellos será encarcelado por 1 año.

Payoff Matrix



The Prisioner Dilemma

- La acción racional individual es *desertar*
  Esto garantiza una utilidad no peor que 2, mientras que cooperar garantiza una utilidad de a lo más 1
- Luego, deserción es la mejor respuesta a todas las posibles estrategias: ambos agentes desertan, y obtienen utilidad 2
- PERO la *intuición* dice que esta no es la mejor salida:
  - Probablemente ambos deberían cooperar y así obtendrían una utilidad de 3.

The Prisioner Dilemma

- Esta supuesta paradoja es *el problema fundamental de las interacciones multiagente*. Esto parece implicar que la cooperación no ocurrirá en sociedades de agentes *auto-motivados*.
- Problemas del mundo real:
  - Asuntos limítrofes entre paises.
  - Tratados de libre comercio.
  - etc.
- ¿Podemos recuperar la cooperación?.

Agreements between Agents

¿Cómo los agente pueden llegar a acuerdo cuando ellos son auto-motivados?

- En muchos escenarios, es posible lograr acuerdos de beneficio mutuo.
- Las capacidades de negociación y *argumentación* son claves para lograr tales acuerdos

Mechanisms and Protocols

- La negociación está controlada por un mecanismo particular: *protocolo*.

- Este mecanismos define las "*rules of encounter*" entre los agentes.
- Propiedades deseables:
  - *Convergencia (éxito garantizado)*
  - *Maximización de bienestar social*
  - *Eficiencia de Pareto*
  - *Racionalidad individual*
  - *Estabilidad*

Bids

- La cooperación se puede llevar a cabo a través de “subastas”.
- Una *subasta* ocurre entre un agente conocido como el *subastador* y un conjunto de agentes conocido como los *apostadores*
- El objetivo de la subasta es que el subastador asigne un *bien* a uno de los apostadores
- En muchos escenarios, el subastador desea maximizar el precio mientras que los apostadores desean minimizarlo

Example: *English Bidding*

- Es el tipo más conocido de subasta:
  - *Primer-precio*
  - *Open-cry (abierta a todos)*
  - *Ascendente*

- La estrategia dominante para el agente implica apostar sucesivamente una pequeña cantidad mayor que la apuesta más alta actual hasta que se alcance su valoración, entonces retirarse.

Negotiation

- Subastas solo se preocupan de la asignación de bienes: se requieren mejores técnicas para lograr acuerdos
- La *Negociación* es el proceso iterativo de lograr acuerdos en temas de interés común.
- Un escenario de negociación tiene 4 componentes:

1. **Un conjunto de negociación (propuestas posibles)**
2. **Un protocolo**
3. **Estrategias privadas (una por cada agente)**
4. **Una regla que determina cuando se ha logrado un trato y cual es el acuerdo.**

Negotiation

- *Efficient*: Eficiencia de Pareto
- *Stable*: No hay incentivo para desviarse
- *Simple*: Bajo costo computacional y de comunicación.
- *Distributed*: No hay toma de decisiones centralizada
- *Simetric*: Agentes toman roles equivalentes.

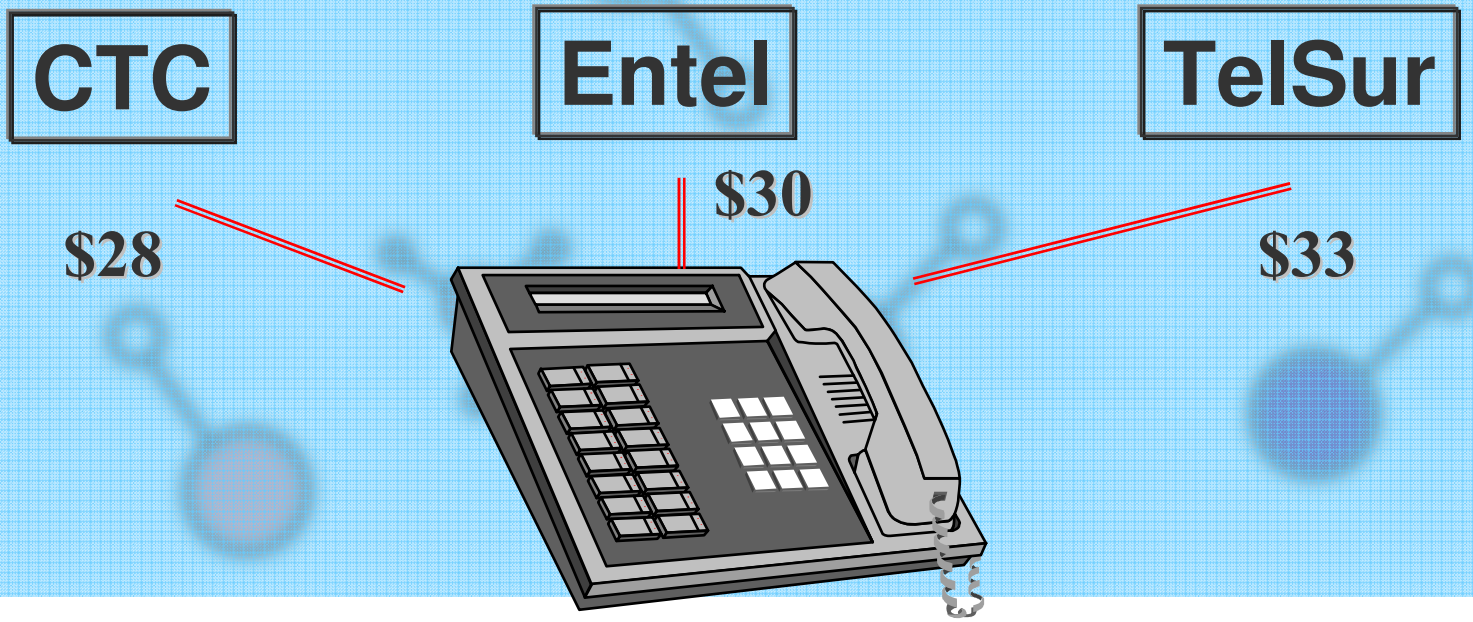
Negotiation Protocol

A negotiation protocol proceeds in rounds:

1) En round 1, los agentes proponen simultáneamente un acuerdo del conjunto de negociación
2) Si un agente encuentra que el trato propuesto por el otro es al menos tan bueno o mejor que su propuesta, entonces se logra un acuerdo.
3) Si no se logra un acuerdo, la negociación procede a otro round de propuestas simultáneas
4) En round $u + 1$, no se permite que los agentes hagan una propuesta que es menos preferida por el otro agente que el trato que este propuso en el instante u
5) Si ninguno de los agentes logra una "concesión" en algún round $u > 0$, la negociación finaliza con el acuerdo de conflicto

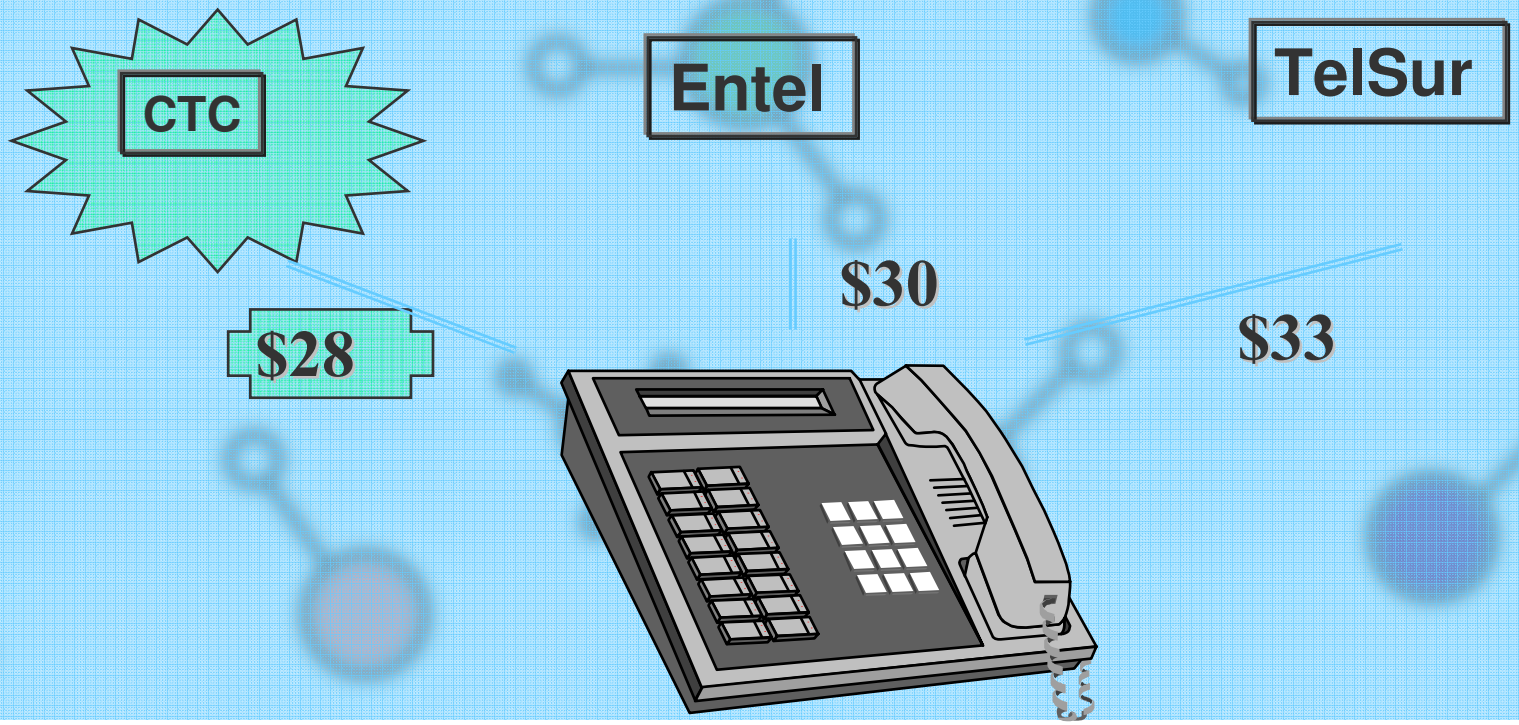
Example: Phone Calls

- Clientes desean hacer llamadas de larga distancia
- “Carriers” apuestan simultáneamente, enviando los precios propuestos
- El teléfono elige automática y dinámicamente el carrier



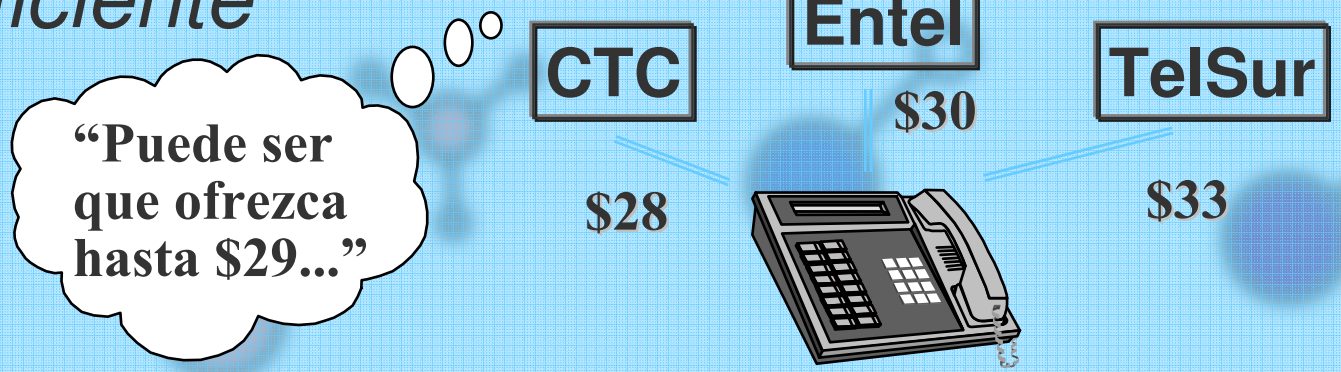
The best bid wins…

- Teléfono elige carrier con la oferta más baja
- Carrier obtiene cantidad que ofreció



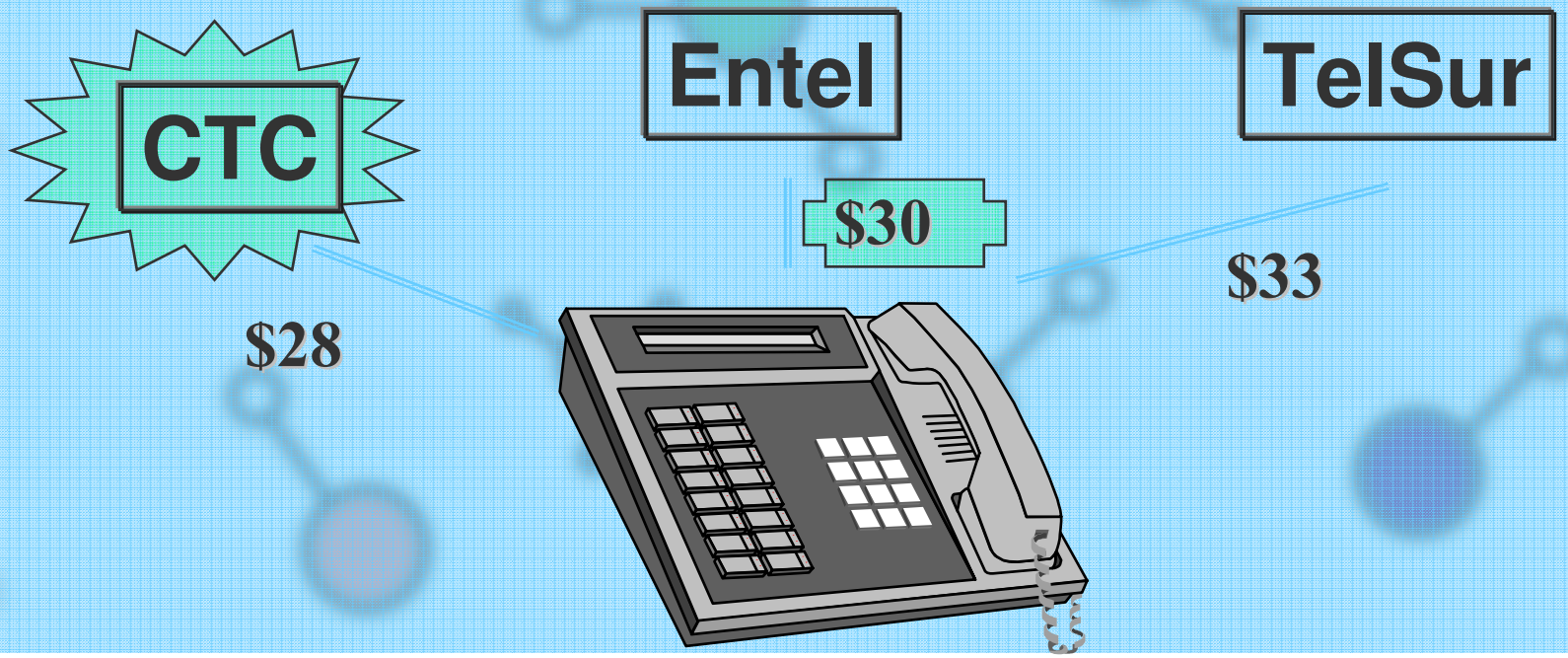
The Mechanism…

- *Distribuido*
- *Simétrico*
- × *Estable*
- × *Simple*
- × *Eficiente*



The best bid wins, gets second price

- Teléfono elige carrier con la oferta más baja
- Carrier obtiene la cantidad del segundo mejor precio.



Communication between Agents

- Necesitamos (re)considerar *macro-aspectos* de la tecnología de agentes inteligentes: aquellos relacionados a la *sociedad,* en vez de lo individual.
- Esto involucra dos aspectos: *Comunicación y Cooperación.*
- *Comunicación basada en Speech Acts.*
- Las teorías de Speech Acts son teorías de uso del lenguaje: intentan dar cuenta de como la gente utiliza el lenguaje diariamente para lograr sus objetivos e intenciones.

Speech Acts

- Algunos mensajes (“oraciones”) son una forma de “acciones físicas” que parecen *cambiar el estado del mundo*
- Ejemplos incluirían:
  - Declarar la guerra
  - Despedir a un empleado
  - ‘Los declaro marido y mujer’ :-)
- Pero más generalmente, *todo* lo que oración es producida con la intención de satisfacer algún objetivo o intención (Acciones del habla).

Speech Acts

- En general, un acto de habla puede verse como dos componentes:
  - Un *verbo (acción) performativo:*
    (ej. Requerir, informar, prometer, …)
  - *Contenido proposicional:*
    (ej. “la puerta está cerrada”)

KQML and KIF

- Ahora consideremos lenguajes para comunicación de agentes o *Agent Communication Languages* (ACLs) — formatos estándar para el intercambio de mensajes.
- El ACL más popular es KQML, desarrollado por la agencia ARPA.

- KQML se compone de dos partes:
  - Un lenguaje de consultas y manipulación de conocimientos (KQML)
  - El formato de intercambio de conocimiento (KIF)

KQML and KIF

- KQML es un lenguaje “exterior” que define varios “acciones comunicativas” aceptables o *performativas*
  Ejemplos de performativas:
  - `ask-if` (‘¿es verdad que… ?’)
  - `perform` (‘por favor realiza la siguiente acción.. ’)
  - `tell` (‘es verdad que… . ’)
  - `reply` (‘la respuesta es . . . ’)
- KIF es un lenguaje para expresar el contenido del mensaje

KIF

- "la temperatura de m1 es 83 grados Celsius":

```
(= (temperature m1) (scalar 83 Celsius))
```

- "cualquier individuo con la propiedad de ser una persona tiene también la propiedad de ser un mamífero":

```
(defrelation persona (?x) :=>
(mamifero ?x))
```

KQML and KIF

- Con el fin de comunicarse, los agentes deben acordar un conjunto común de términos.
- Una especificación formal de un set de términos se denomina una *ontología*
- Ejemplo de diálogo KQML/KIF:

```
A to B: (ask-if (> (size chip1) (size chip2)))
B to A: (reply true)
B to A: (inform (= (size chip1) 20))
B to A: (inform (= (size chip2) 18))
```

Conclusions

- ✓ La Computación basada en Agentes y Sistemas Multi-agente representan un nuevo paradigma para el desarrollo de sistemas complejos en ambientes altamente dinámicos.
- ✓ Agentes como nuevo paradigma de Ingeniería de Software que destaca la interacción como pilar fundamental de la conducta inteligente compleja.
- ✓ La complejidad de los SMA resulta de la forma en que se interconectan los agentes y NO en como se diseñan.
- ✓ Agentes NO son:
  - o Módulos especializados de software tradicional.
  - o Sistemas expertos.