



JASON

Ambiente Livre para Sistemas Multiagentes
Inteligência Artificial

Alexandre Zamberlan

alexz@ufn.edu.br

Laboratório de Práticas de Computação

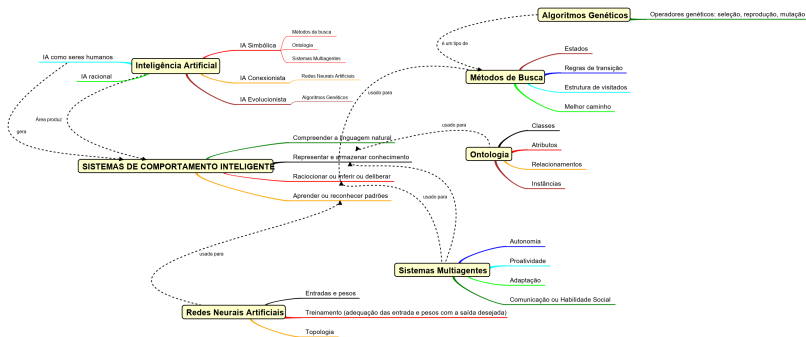
Agenda

- 1 Contexto
- 2 Sistemas de Comportamento Inteligente
- 3 Sistemas Multiagentes
- 4 Considerações

Agenda

- 1 Contexto
- 2 Sistemas de Comportamento Inteligente
- 3 Sistemas Multiagentes
- 4 Considerações

Mapa Mental da IA



Revolução atual: IA é a nova eletricidade

Algumas aplicações de IA



Algumas aplicações de IA



Agenda

- 1 Contexto
- 2 Sistemas de Comportamento Inteligente**
- 3 Sistemas Multiagentes
- 4 Considerações

Sistemas de Comportamento Inteligente

- Deve ser capaz de adaptar-se a novas situações
- Raciocinar relações entre fatos
- Descobrir significados
- Reconhecer e aprender com base em experiências

Agenda

- 1 Contexto
- 2 Sistemas de Comportamento Inteligente
- 3 Sistemas Multiagentes**
- 4 Considerações

Sistemas Multiagentes



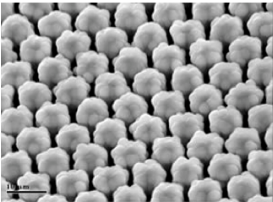
(A)



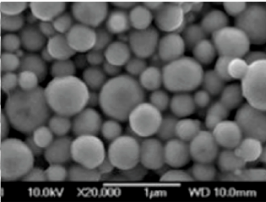
(B)



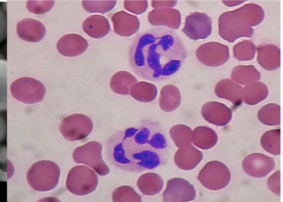
(C)



(D)

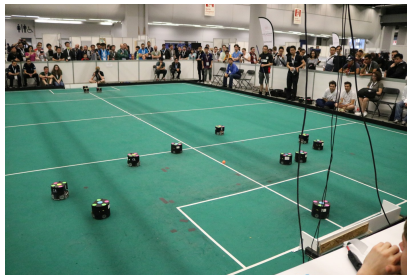


(E)



(F)

Sistemas Multiagentes



Sistemas Multiagentes

- Método baseado em comportamento coletivo
- Agentes como elementos centrais
 - autônomos
 - flexíveis e adaptáveis
 - reativos: percebem e atuam (ambiente)
 - habilidade social: interagem entre si
- **Metodologias e ferramentas consolidadas**
- Abordagem permite grau de abstração elevado
 - simulações \implies maior número de variáveis, restrições, operações, exceções
 - *Bottom-up* e *Top-down* \implies como sistemas particulados

[Bordini and Hübner 2009]

Kits ou ferramentas de desenvolvimento

Possuem uma variedade de características e funcionalidades:

- ambiente de desenvolvimento integrado;
- linguagem de programação;
- sistema operacional;
- suporte ao usuário (manuais e exemplos);
- integração com outras bibliotecas;
- possibilidade de executar o sistema com visualização 2D e 3D;
- propriedade de visualização de cenários de simulação.

Principais ferramentas

FLAME, **JASON**, MASON, NetLogo, Repast, SeSAm, SWARM

JASON: Kit De desenvolvimento SW Livre e Aberto



The screenshot shows the JASON website. At the top left is a classical painting of a figure. To its right, the text reads: "Jason a Java-based interpreter for an extended version of AgentSpeak". Below this is a navigation bar with links: Home, Description, Documents, Examples, Demos, Teaching, Projects, and a search bar. The main content area features the JASON logo (a classical head) and the text "About Jason". It describes JASON as an interpreter for an extended version of AgentSpeak, implementing operational semantics and providing a platform for multi-agent systems. It is available as Open Source under GNU LGPL. Below this are sections for "Links" (pointing to GitHub, Sourceforge, and Screenshots) and "Authors" (listing Jomi F. Hübner and Rafael H. Bordini). On the right side, there are buttons for "Download the latest version of Jason!" and "ECLIPSE PLUG-IN". A "News" section dated 25/08/2017 mentions the Multi-Agent Programming Contest 2017 and lists winners.

Jason
a Java-based interpreter for an
extended version of AgentSpeak

Jason
by Gustavo Moreira (2002)
Oil on canvas, 204 x 113 cm
Musée d'Orsay, Paris
© Photo 2004, Photograph by
Hervé Lecomte

Home Description Documents Examples Demos Teaching Projects Search



About Jason

Jason is an interpreter for an extended version of AgentSpeak. It implements the operational semantics of that language, and provides a platform for the development of multi-agent systems, with many user-customisable features. **Jason** is available Open Source, and is distributed under GNU LGPL. See more in the [Description](#) page.

Links

- Jason on [Github](#) (latest code);
- Jason on [Sourceforge](#);
- [Screenshots](#).

Authors

Jason is developed by [Jomi F. Hübner](#) and [Rafael H. Bordini](#), based on previous work done with many colleagues, in particular Michael Fisher, Joyce Martins, Álvaro Moreira, Renata Vieira, Willem Visser, Mike Wooldridge, but also many others, as acknowledged in

Download the latest
version of Jason!

 **DOWNLOAD**

Read the tutorial for installing
Jason as an Eclipse plug-in!

 **ECLIPSE PLUG-IN**

News

25/08/2017
[The Multi-Agent Programming Contest 2017](#) had two teams using **Jason**:

- 2nd place: Jason-DTU
- 4th place: SMART-JaCaMo (using [JaCaMo](#))

Watch replays of the matches and
see the [results here](#).

20/08/2017

[Bordini and Hübner 2009]

JASON: Modelagem

- evento ativador (*trigger*)
- contexto ou condição
- plano ou planos contingenciais
- atualização da base de conhecimento (fatos/crenças)
- Vídeo no Youtube sobre modelagem de Agentes
 - https://youtu.be/IBNh7j_F4yc

JASON: Modelagem

- Unidade básica é o AGENTE
 - sensores
 - conjunto de planos
 - atuadores
- Características fundamentais:
 - Autonomia - *threads*
 - Proatividade - planos/métodos sobrecarregados
 - Adaptação ou flexibilidade - tratamento de exceções
 - Comunicação - *socket*
 - enviar um fato/crença
 - enviar um questionamento/pergunta
 - enviar uma ação ou plano

JASON: arquivos

```
1 Jason - três tipos de arquivo
2
3 .mas2j
4     -> descrição do projeto do SMA
5
6
7 .java
8     -> código java que implementa o ambiente em que os agentes estão inseridos
9     -> principais métodos:
10         init() -> inicialização do SMA -> com percepções iniciais
11         executeAction() -> tratamento das ações solicitadas pelos agentes
12
13     -> métodos secundários:
14         addPercept() -> ambiente avisando todos os agentes de uma percepção
15         removePercept() -> ambiente retirando uma percepção gerada
16
17
18 .asl
19     -> código AgentSpeak(L), que é um Prolog melhorado, contendo:
20         - crenças ou fatos do ambiente (base de conhecimento de um agente)
21         - planos ou regras (conjunto de ações que são disparadas quando um evento
22           ocorre no ambiente - evento externo)
23         - subplano (conjunto de ações que são disparadas quando um evento interno
24           ocorre)
25     -> alguns comandos de apoio que são do Jason
26         - .print() -> exibe alguma coisa no terminal de verbose
27         - .send() -> diretiva de comunicação entre agentes
28             .send(agente,tell,crenca/fato)
29             .send(agente,achieve,plano)
30             .send(agente,untell,crenca/fato)
31
32         - .broadcast -> diretiva de comunicação de um agente para muitos agentes
33             .broadcast(tell,crenca/fato)
34             .broadcast(untell,crenca/fato)
35
36         - .wait(milissegundos)
```

JASON: .mas2j

```
1  MAS almoxarifado {  
2  
3      infrastructure: Centralised  
4  
5      environment: Entrada  
6  
7      agents:  
8          r2;  
9          r1;  
10  
11     aslSourcePath:  
12         "src/asl";  
13 }
```

JASON: .asl r1

```
1 // Agent r1 in project almoarifado
2 /* Initial beliefs and rules */
3 viagens(3).
4
5 /* Plans */
6 +peca(grande) : viagens(Qtd) & Qtd > 0
7     <- .print("percebi uma peca grande e vou guarda-la");
8     V = Qtd - 1;
9     -viagens(Qtd);
10    +viagens(V);
11    .print("tenho mais ", V, " viagens");
12    guardarPecaGrande.
13
14 +peca(grande) : true
15     <- .print("percebi uma peca grande mas não posso mais guardar... vamos empilhar");
16     empilharPecaGrande.
17
18 +peca(media) : viagens(Qtd) & Qtd > 0
19     <- .print("percebi uma peca media e vou guarda-la");
20     V = Qtd - 1;
21     -viagens(Qtd);
22     +viagens(V);
23     .print("tenho mais ", V, " viagens");
24     guardarPecaMedia.
25
26 +peca(media) : true
27     <- .print("percebi uma peca media mas não posso mais guardar... vou chamar r2");
28     .send(r2,achieve,peca(media)).
29
30 +chegou(gasolina) : viagens(Qtd) & Qtd == 0
31     <- -viagens(0);
32     +viagens(3);
33     abastecer.
```

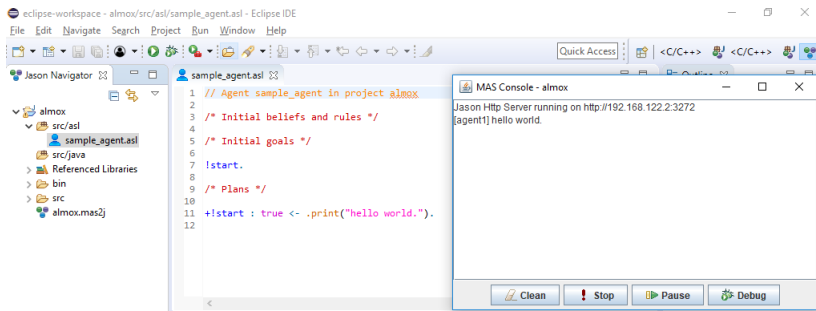
JASON: .asl r2

```
1 // Agent r2 in project almojarifado
2
3 /* Plans */
4 +!peca(media)[source(Agt)] : true
5     <-
6         .print(Agt," me mandou guardar peça média");
7         guardarPecaMedia.
8
9 +peca(pequena) : true
10     <- .print("percebi uma peça pequena e vou guarda-la");
11     guardarPecaPequena.
12
```

JASON: .java environment

```
21 @Override
22 public void init(String[] args) {
23     super.init(args);
24     addPercept(Literal.parseLiteral("peca(media)"));
25 }
26
27 @Override
28 public boolean executeAction(String agName, Structure action) {
29     if (action.getFunction().equals("guardarPecaGrande")) {
30         System.out.println("[ambiente] " + agName + " está guardando peca grande");
31         removePercept(Literal.parseLiteral("peca(grande)"));
32     } else if (action.getFunction().equals("empilharPecaGrande")) {
33         contatEmpilhamentos++;
34         System.out.println("[ambiente] mais uma peca grande empilhada na entrada: " + contatEmpilhamentos);
35         removePercept(Literal.parseLiteral("peca(grande)"));
36     } else if (action.getFunction().equals("guardarPecaMedia")) {
37         System.out.println("[ambiente] " + agName + " está guardando peca média");
38         removePercept(Literal.parseLiteral("peca(media)"));
39     } else if (action.getFunction().equals("guardarPecaPequena")) {
40         System.out.println("[ambiente] " + agName + " está guardando peca pequena");
41         removePercept(Literal.parseLiteral("peca(pequena)"));
42     } else if (action.getFunction().equals("abastecer")) {
43         System.out.println("[ambiente] " + agName + " está reabastecendo");
44         removePercept(Literal.parseLiteral("chegou(gasolina)"));
45     } else logger.info("tentando executar " + action + ", mas ainda não implementado!");
46
47     try {
48         Thread.sleep(2000);
49     } catch (InterruptedException e) {
50         // TODO Auto-generated catch block
51         e.printStackTrace();
52     }
53     if (contatEmpilhamentos == 3) {
54         addPercept(Literal.parseLiteral("chegou(gasolina)"));
55         contatEmpilhamentos = 0;
56     }
57     addPercept(Literal.parseLiteral("sorteiaPeca()"));
58
59     return true; // the action was executed with success
60 }
```


JASON no Eclipse














Agenda


- 1 Contexto
- 2 Sistemas de Comportamento Inteligente
- 3 Sistemas Multiagentes
- 4 Considerações**

GitHub






 alexandreزامberlan / sistemasMultiagentes






 Unwatch 1  Star 0  Fork 0


 Code  Issues  Pull requests  Projects  Wiki  Security  Insights  Settings







Códigos em AgentSpeak e Jason para exemplificar SMA do paradigma BDI 



[Manage topics](#)

 7 commits  1 branch  0 releases  1 contributor  GPL-3.0

Branch: master     

 Laboratório 06 sma almoz de 2019-2 Latest commit 14555b9 9 days ago

 almoxarifado_2019-1	sma almoz de 2019-2	9 days ago
 almoxarifado_2019-2	sma almoz de 2019-2	9 days ago
 docs	material de apoio	23 days ago
 LICENSE	Initial commit	5 months ago
 README.md	Update README.md	5 months ago
 SMA_exemplos.zip	material de apoio	23 days ago

 README.md 

Programação Sistemas Multiagentes

Códigos em AgentSpeak(L) e Jason para exemplificar SMA do paradigma BDI. Exemplos discutidos na disciplina Inteligência Artificial da UFN.

Oportunidades de estudo



- Disciplinas:
 - Programação Paralela e Distribuída
 - Sistemas Distribuídos
 - Inteligência Artificial
- Projetos de pesquisa e extensão

Referências



Bordini, R. H. and Hübner, J. F. (2009).

Agent-based simulation using bdi programming in jason.

In *M. Uhrmacher ; Danny Weyns. (Org.). Multi-Agent Systems: Simulation and Applications, Modeling and Simulaton*, pages 451–476. CRC Press.



Zamberlan, A. O., Kurtz, G. C., Bordini, R. H., and Solan (2015).

Ferramentas de simulação multiagentes no contexto de sistemas nanoparticulados.

In *Olhares Sustentáveis em Favor da Vida*, Santa Maria - RS. XIX SEPE - Simpósio de Ensino, Pesquisa e Extensão, Centro Universitário Franciscano.

JASON

Ambiente Livre para Sistemas Multiagentes

Inteligência Artificial

Alexandre Zamberlan

alexz@ufn.edu.br

Laboratório de Práticas de Computação

OBRIGADO