Escola Politécnica da USP - Depto. de Enga. Mecatrônica

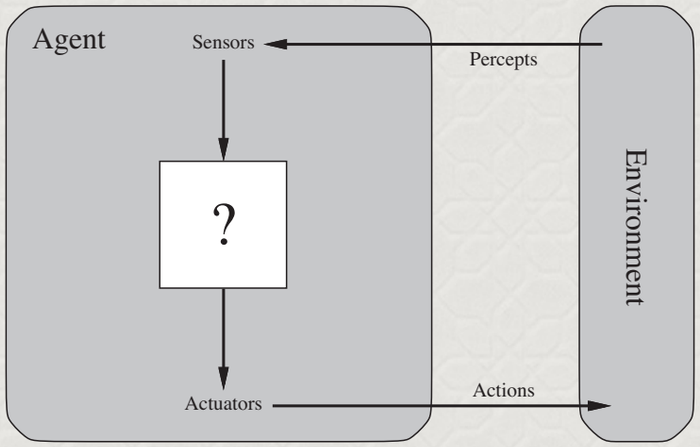
PMR-3510 Inteligência Artificial

Aula 15- Novas e velhas aplicações da IA

Prof. José Reinaldo Silva

reinaldo@usp.br

IA clássica: validando sentenças e seq. de ações



Na aproximação clássica reduzimos as ações a um agente único que vê um "mundo" fechado onde tudo que existe para ser conhecido está na sua base de conhecimento.

Regras

Sofisticando a base de conhecimento

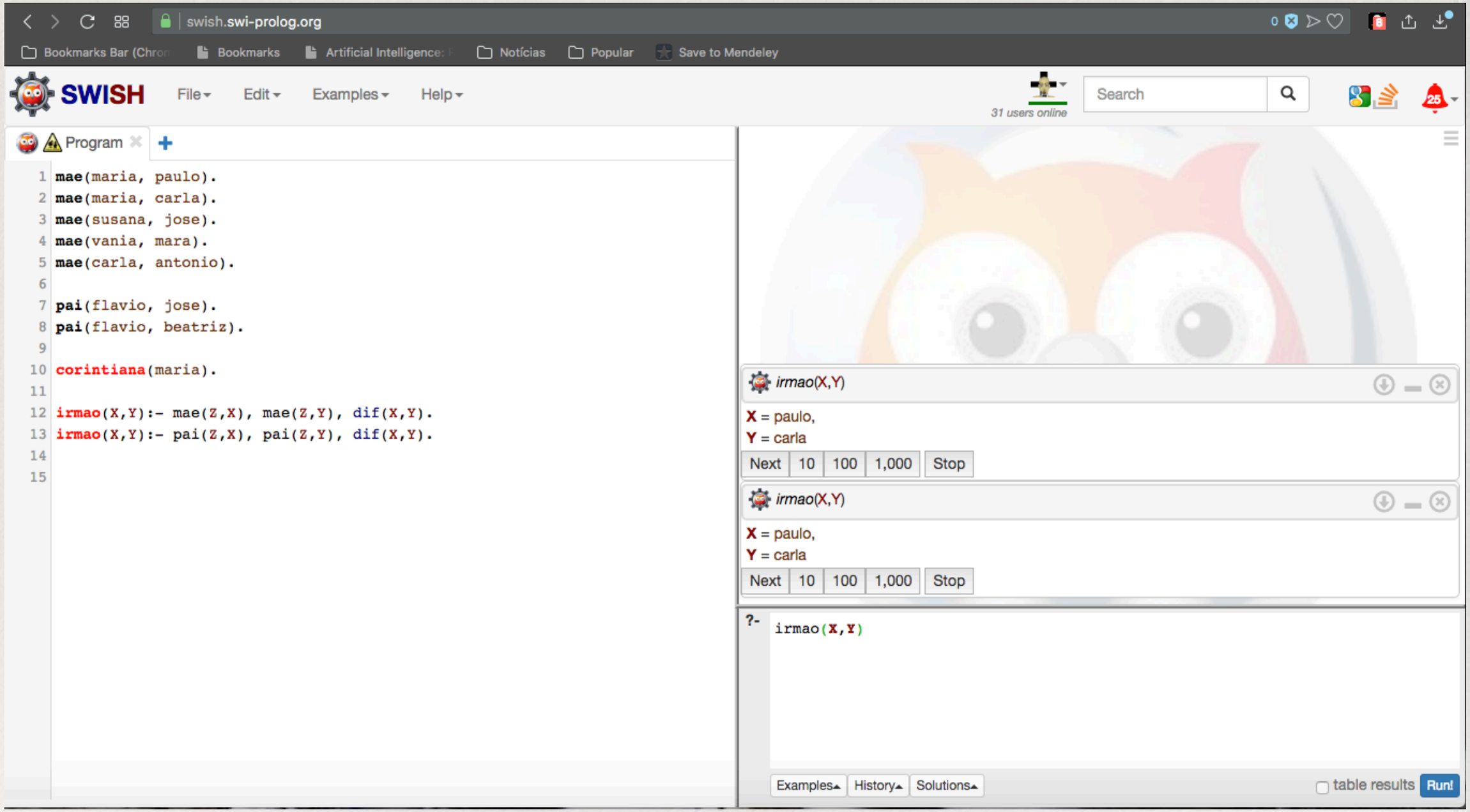
Vamos dar um passo adiante e pensar em relacionar fatos, ou, em outras palavras, ou criar novos fatos (deduzir) à partir de fatos já existentes e de regras de relacionamento. Por exemplo, na nossa base familiar vamos definir o que é o irmão ou irmã. Teoricamente é aquele ou aquela que tem um dos pais em comum, para se bem abrangente. Na nossa base vamos inserir o "conceito" de irmão. E vamos precisar de uma relação pre-definida do Prolog.

Base de conhecimento

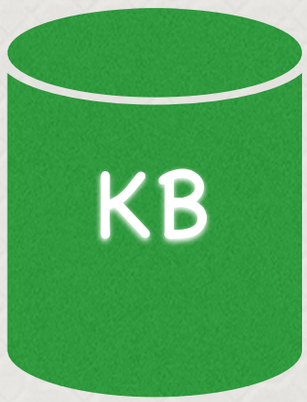
```
mae(maria, paulo).
mae(maria, carla).
mae(susana, jose).
mae(vania, mara).
mae(carla, antonio).

pai(flavio, jose).
pai(flavio, beatriz).
```

```
corintiana(maria)

irmao(X, Y) :- mae(Z, X), mae(Z, Y), dif(X, Y)

irmao(X, Y) :- pai(Z, X), pai(Z, Y), dif(X, Y)
```

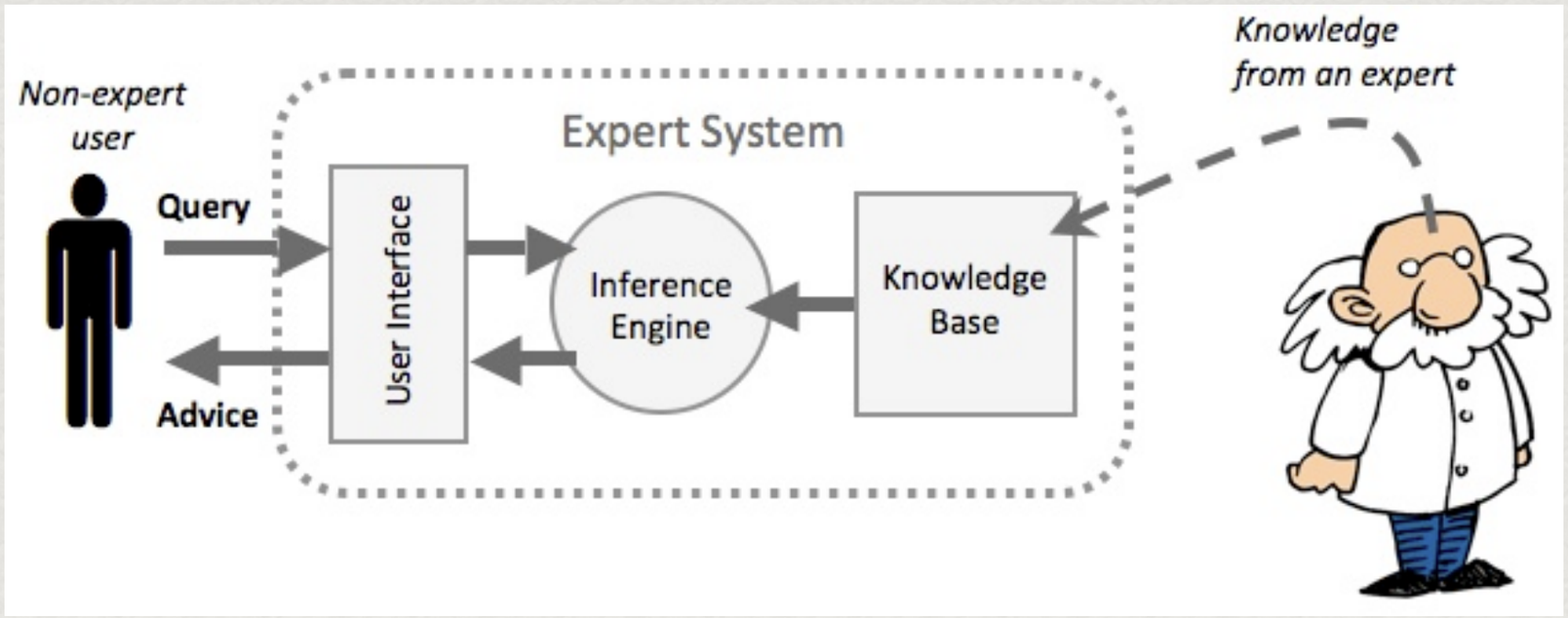
swish.swi-prolog.org



A base de fatos e regras é escrita na forma de termos e cláusulas de Horn. "Queries" podem ser enviados a uma "máquina de inferência" que, usando backward chaining deve mostrar se este "query" q é tal que $KB \models q$



Aplicações



sistemas de diagnóstico (médico, mecânico, etc.), operações náuticas, análise de crédito bancário, perfil de clientes, etc.

Programação Lógica

ProLog

The imperative approach:

1. Enter the coffee shop
2. Queue in the line and wait for the barista asking you for your order
3. Order
4. Yes, for takeaway, please
5. Pay
6. Present your loyalty card to collect points
7. Take your order and walk out

The declarative approach:

1. A large latte for takeaway, please

Forward and backward chaining

Horn Form (restricted)

KB = conjunction of Horn clauses

Horn clause =

- ♢ proposition symbol; or
- ♢ (conjunction of symbols) $\Rightarrow$ symbol

E.g., $C \wedge (B \Rightarrow A) \wedge (C \wedge D \Rightarrow B)$

Modus Ponens (for Horn Form): complete for Horn KBs

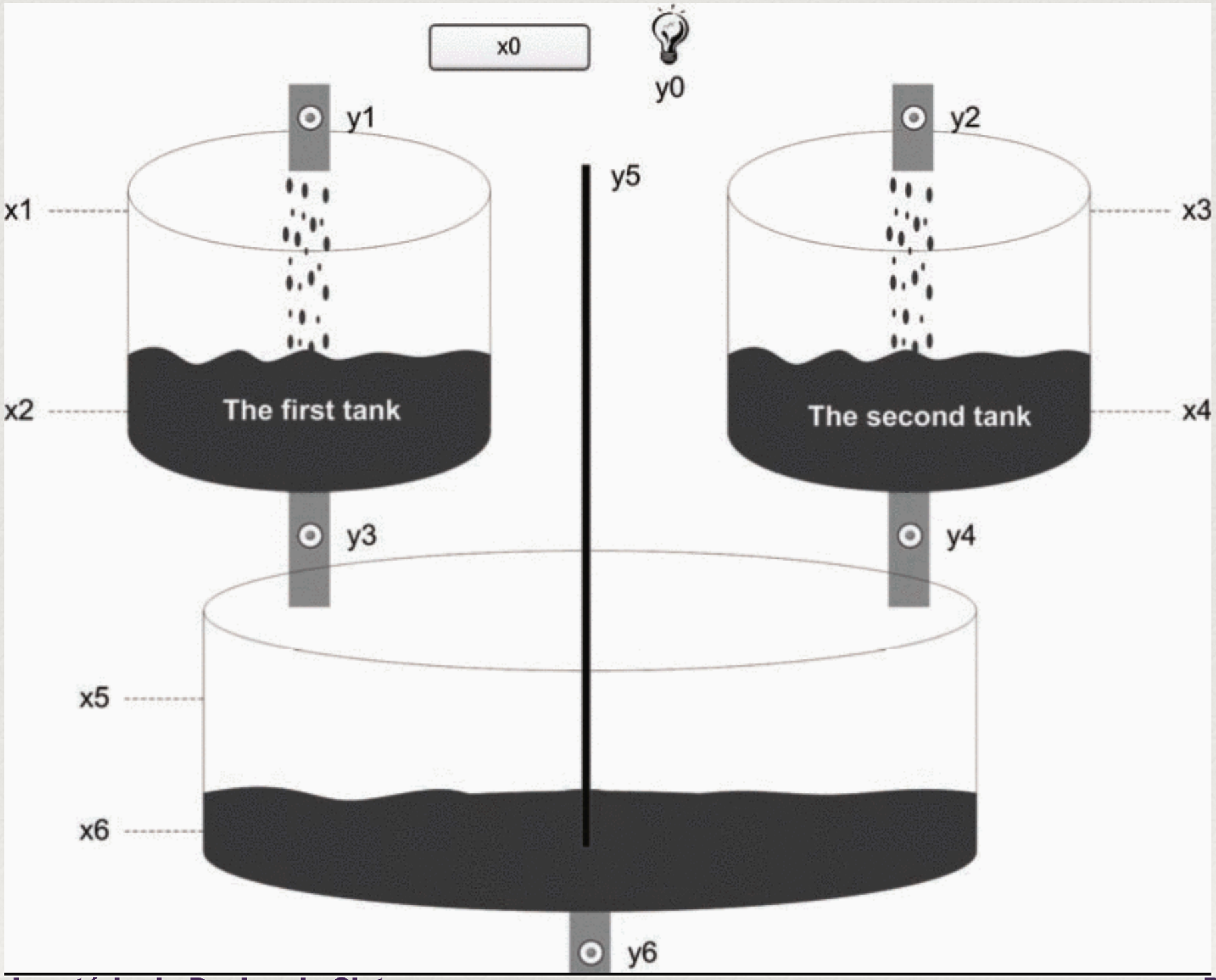
$$\frac{\alpha_1, \ldots, \alpha_n, \qquad \alpha_1 \wedge \cdots \wedge \alpha_n \Rightarrow \beta}{\beta}$$

Can be used with forward chaining or backward chaining.
These algorithms are very natural and run in linear time

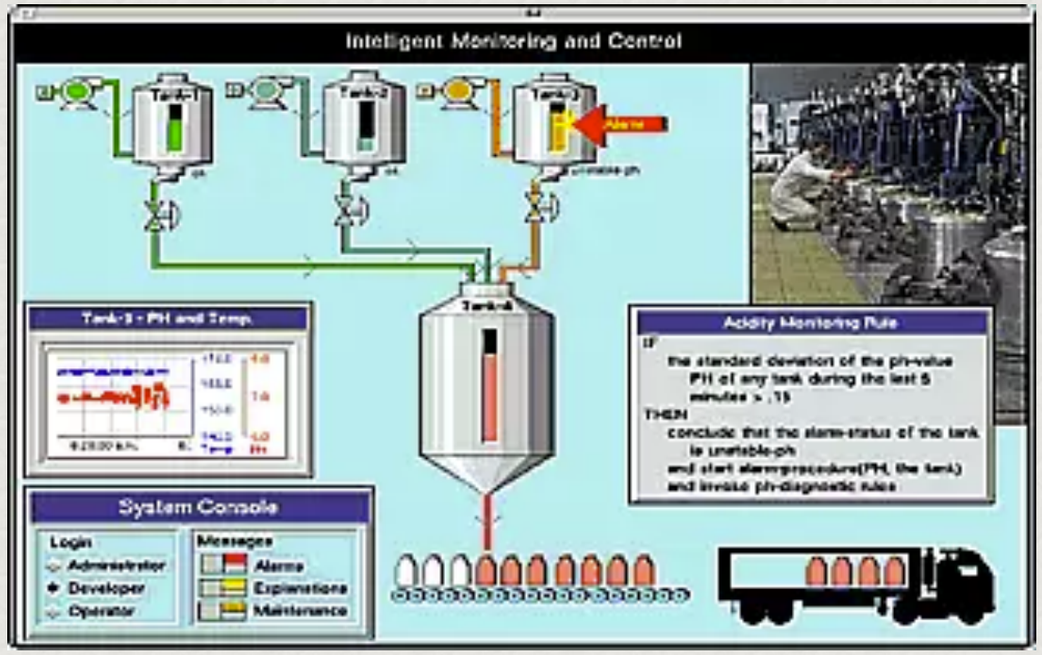
Intelligent Fire Alarm: forward chaining with multiple actions:

1. If smoky is detected go to alarm_mode_1
2. If in alarm_mode_1 then {check(temp1), wait(10), check(temp2) and if temp2 > temp1 go to alarm_mode_2}
3. If in alarm_mode_2 then if visual_detect(flames) go to alarm_mode_3
4. If in alarm_mode_3 then {turn_on(sprinklers), close(elevators), close(cutting_fire_doors), sound_alerts, call(fire_force)}.

Stuart Russell, Univ. of California - Berkeley, autor do livro texto.


x0
y0
y1
y2
y5
x1
x3
x2
x4
The first tank
The second tank
y3
y4
x5
x6
y6

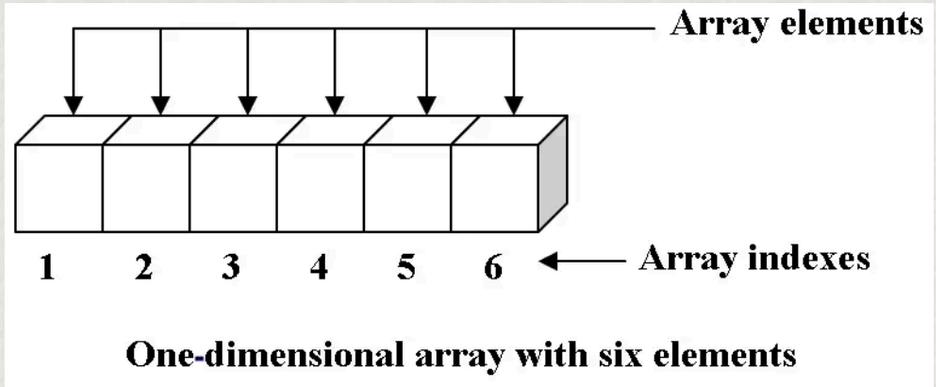

Ignite Technologies
Information Technology Solutions
Gensym G2
The World's Leading Software Platform for Real-Time Expert System Applications


Intelligent Monitoring and Control
Tank-1
Tank-2
Tank-3
Alarm
Tank-3 - PH and Temp.
Acidity Monitoring Rule
IF
the standard deviation of the ph-value
PH of any tank during the last 5
minutes > .15
THEN
conclude that the alarm-status of the tank
is unstable-ph
and start alarm-procedure(PH, the tank)
and invoke ph-diagnostic rules
System Console
Login
Administrator
Developer
Operator
Messages
Alarms
Explanations
Maintenance

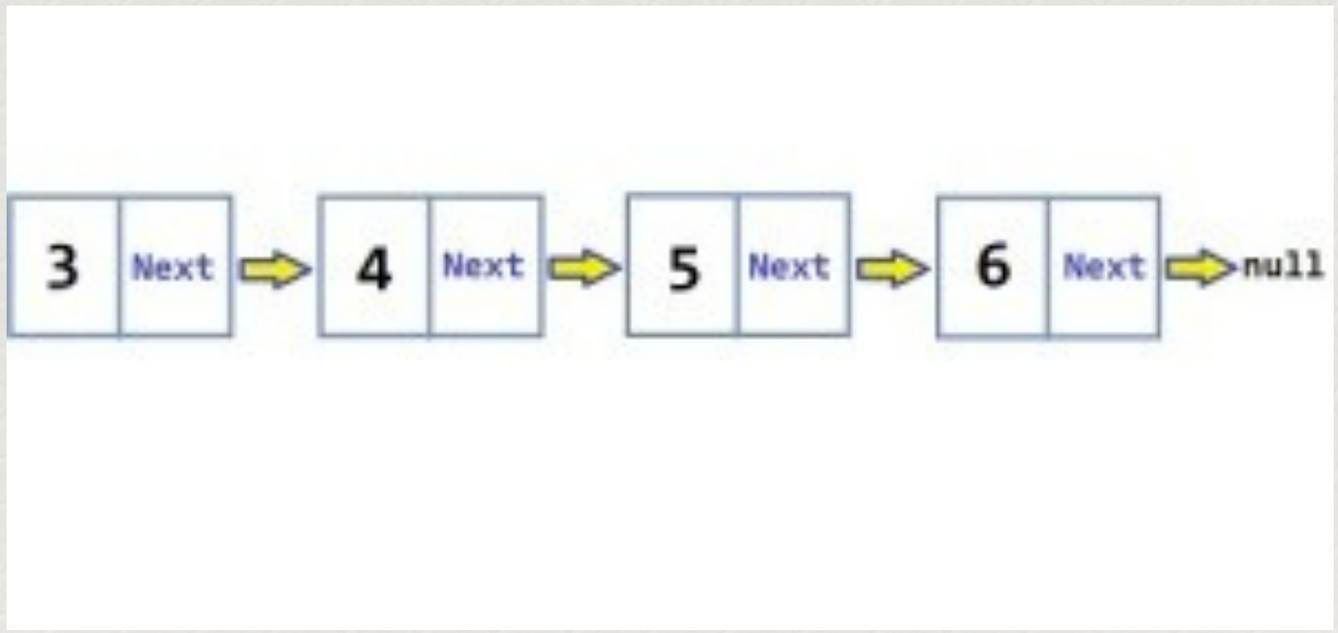

LAFARGE

Mas vários dos processos da IA clássica são baseados em algoritmos de busca sobre estruturas como árvores e grafos. Estas estruturas podem ser implementadas em Prolog (ou em LISP) em estruturas de dados específicas como as listas.

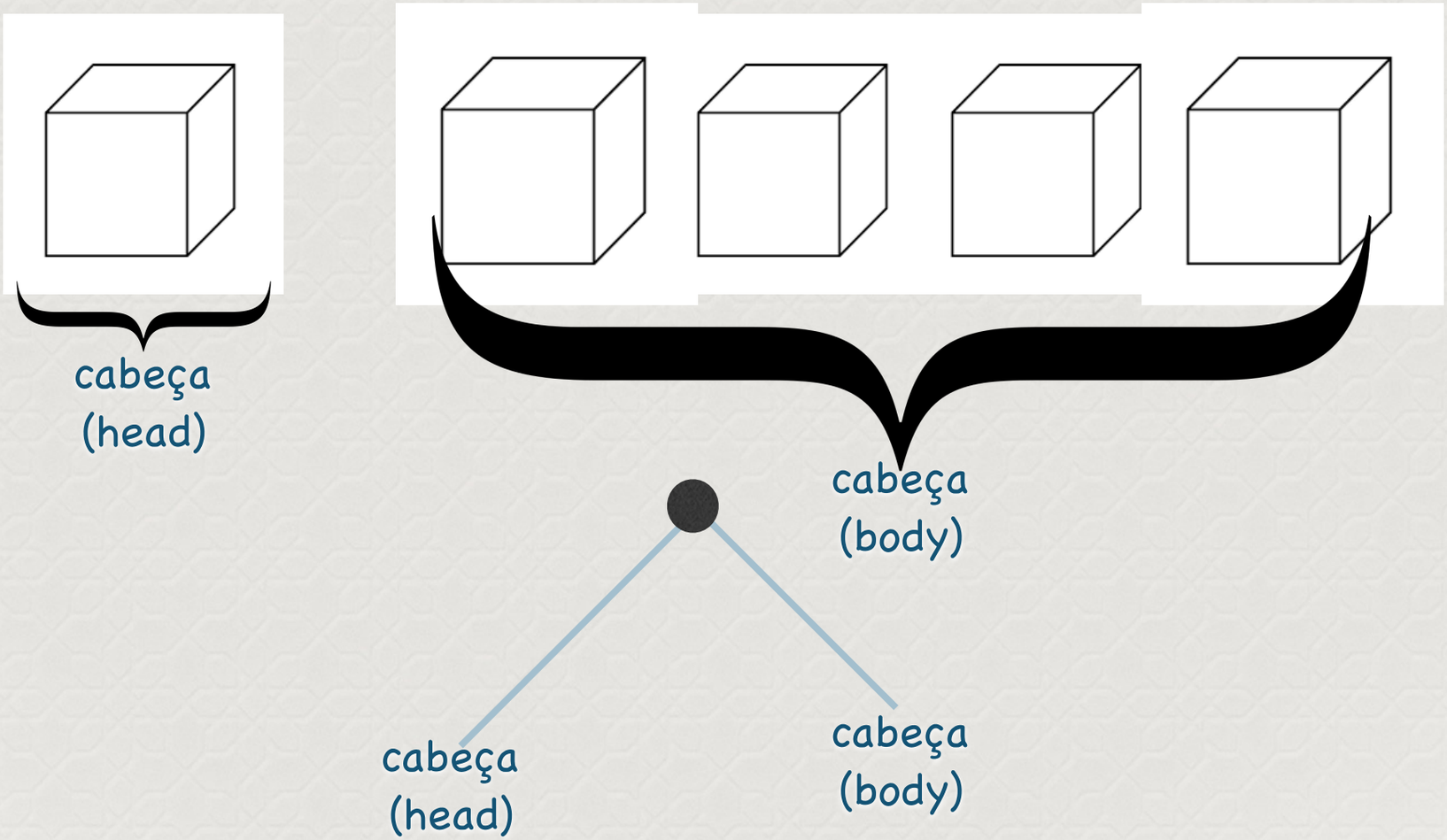
Instancias da estrutura abstrata de lista



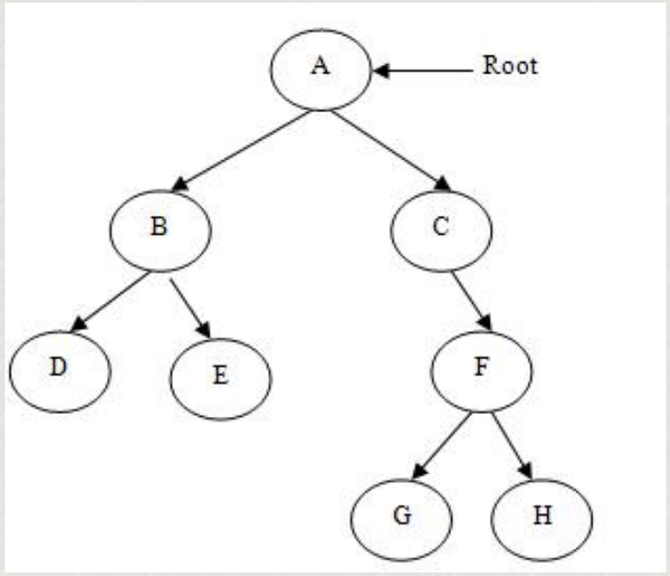
One-dimensional array with six elements

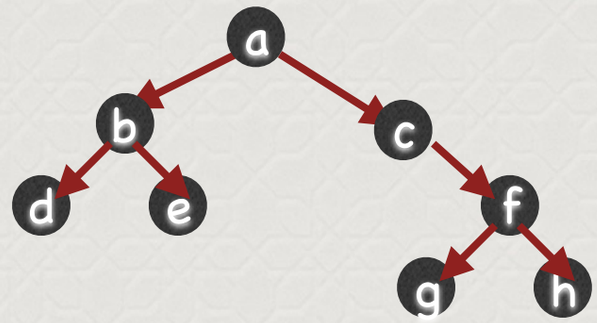


Estrutura abstrata de lista



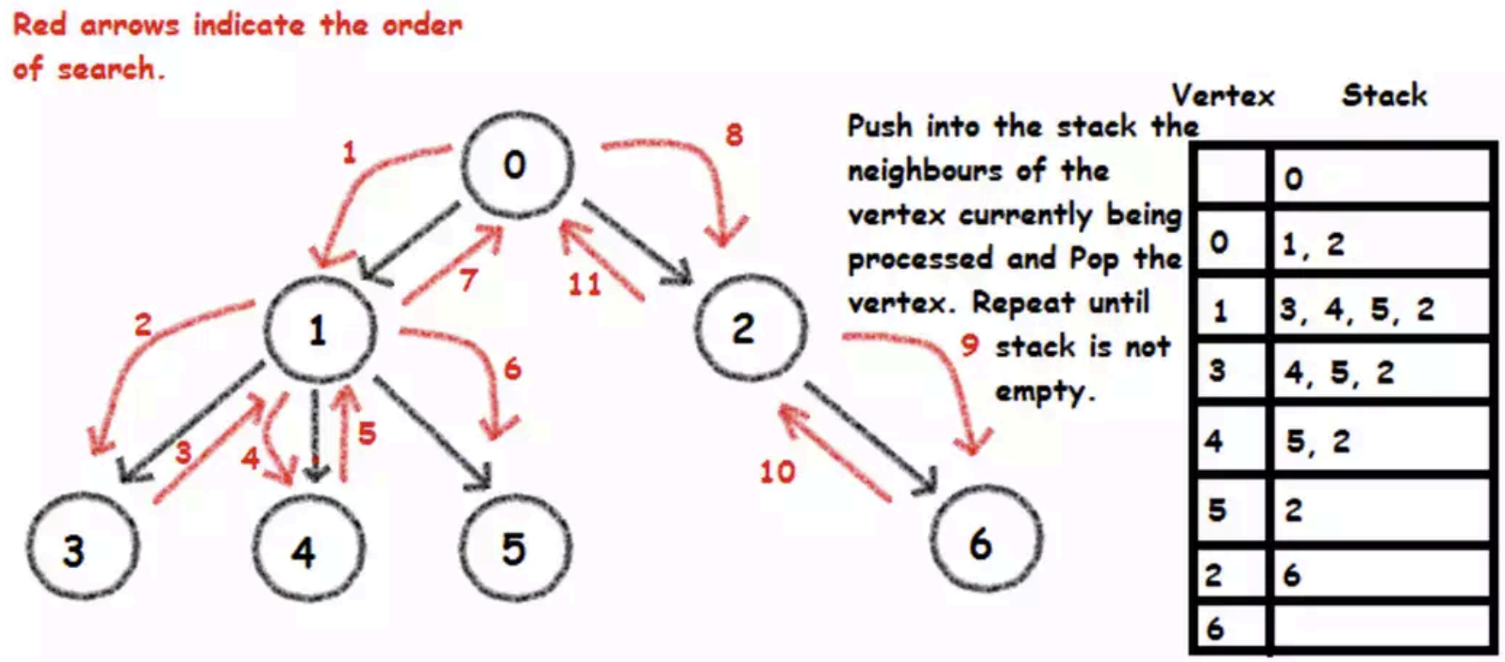
Portanto é possível utilizar uma lista para implementar uma estrutura de árvore





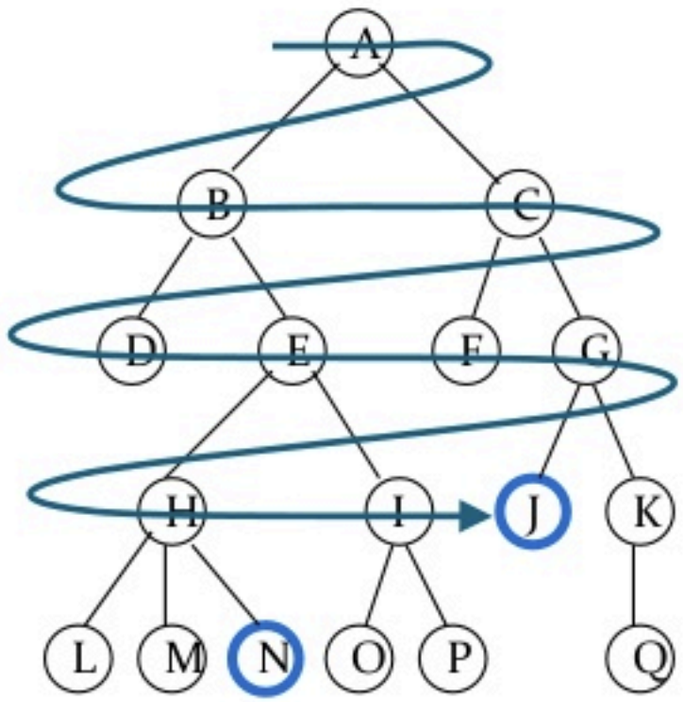
[a, [b, [d, e]], [c, [f, [g, h]]]]

Busca em profundidade



Busca em largura

Breadth-first searching[1]



- A breadth-first search (BFS) explores nodes nearest the root before exploring nodes further away
- For example, after searching A, then B, then C, the search proceeds with D, E, F, G
- Node are explored in the order A B C D E F G H I J K L M N O P Q
- J will be found before N

Best-first Search Algorithms

There are of course many different ways of defining a heuristic function h. But there are also different ways of *using* h to decide which path to expand next; which gives rise to different best-first search algorithms.

One option is *greedy* best-first search:

- expand a path with an end node n such that $h(n)$ is *minimal*

Breadth-first and depth-first search may also be seen as special cases of best-first search (which do not use h at all):

- Breadth-first: expand the (leftmost of the) *shortest* path(s)
- Depth-first: expand the (leftmost of the) *longest* path(s)

The A* Algorithm

The central idea in the so-called *A* algorithm* is to guide best-first search both by

- the estimate to the goal as given by the heuristic function h and
- the cost of the path developed so far.

Let n be a node, $g(n)$ the cost of moving from the initial node to n along the current path, and $h(n)$ the estimated cost of reaching a goal node from n. Define $f(n)$ as follows:

$$f(n) \quad = \quad g(n) + h(n)$$

This is the *estimated cost of the cheapest path through* n leading from the initial node to a goal node. *A* is the best-first search algorithm that always expands a node* n *such that* $f(n)$ *is minimal.*

Estratégias de Busca Informada

Análise de custo

$f(x) = g(x)$

Heurística

$f(x) = h(x)$

Algoritmo A*

$f(x) = g(x) + h(x)$

Tipos de problema resolvidos por IA

Podemos classificar os problemas aos quais aplicamos (com sucesso IA em:

i) completamente reativos
ii) interativos (jogos, sistemas especialistas, apoio a decisão)
iii) preditivos (planning, scheduling, etc.)
iv) evolutivos (aprendizado de máquina)

Domain Specific Planning (DSP)
X
Domain Independent Planning (DIP)

In DSP the idea is to use specific properties of the problem domain, besides the problem definition, to synthesize a plan.

In DIP the goal is to find out generic ways to deal with the problem definition, no matter which is the problem being solved.

Base conceitual pra o problema de planning (I)

Um "sistema de planejamento" é composto de um problema de planejamento (estado inicial, estado final, e conjunto de ações), da descrição do ambiente onde o plano deverá ser realizado.

STRIPS: Stanford Research Institute Problem Solver

Em 1971, Richard Fikes e Nils Nilsson, propuseram um algoritmo geral para resolução de problemas em IA que ficou conhecido como STRIPS, em uma alusão ao SRI, onde a pesquisa foi realizada. Mais tarde STRIPS passou a significar também uma linguagem para tratar problemas de planejamento. Foi substituida por outras mas todas inspiradas no algoritmo e na linguagem original.

Base conceitual para o problema de planning (III)

Um sistema estado-transição é uma 4-upla $\Sigma = (S, A, E, \gamma)$ onde:

- S é um conjunto de finito (ou recursivamente enumerável) de estados;
- A é um conjunto finito (ou recursivamente enumerável) de ações;
- E é um conjunto finito (ou recursivamente enumerável) de eventos;
- $\gamma = (S \times A \times E) \rightarrow 2^S$ é a função de transição.

Ghallab, M., Nau, D., Traverso, P.; Automated Planning: Theory and Practice, Morgan Kaufmann, 2004

Algumas limitações:

- a primeira limitação é que vamos supor que temos problemas de planejamento determinísticos: as ações produzem sempre o mesmo efeito, são sempre as mesmas e agem uma de cada vez;
- as ações não causam nenhum efeito acumulativo e portanto vale a aproximação já feita vale: o problema de planning pode ser formalizado como um sistema dinâmico discreto e abordado com um sistema estado-transição;
- uma vez delimitado o problema de planejamento este não está sujeito a nenhuma influência externa.

www.cs.cmu.edu/~avrim/graphplan.html

Bookmarks Bar (Chro… | Bookmarks | Artificial Intelligence: … | Notícias | Popular | Save to Mendeley

Welcome to the Graphplan Home Page

School of Computer Science, Carnegie Mellon University, Pittsburgh PA 15213-3891
People: Avrim Blum, Merrick Furst, John Langford

About Graphplan

Graphplan is a general-purpose planner for STRIPS-style domains, based on ideas used in graph algorithms. Given a problem statement, Graphplan explicitly constructs and annotates a compact structure called a Planning Graph, in which a plan is a kind of "flow" of truth-values through the graph. This graph has the property that useful information for constraining search can quickly be propagated through the graph as it is being built. Graphplan then exploits this information in the search for a plan. Graphplan was created by Avrim Blum and Merrick Furst, with subsequent extensions and improvements made by many researchers at many different institutions around the world.

How to try it out

To try out Graphplan, go to the Graphplan home directory. This directory contains source code, object code for the DECstation and Sparcstation, a variety of sample domains, and a README file that describes how to run Graphplan and how to make up your own domains and problems. The program allows you to see an **animation** (in X) of what it's doing. For instance, look at a simple TSP domain. Here is what the animation looks like on a more interesting Flat Tire World (fixit) domain (graph creation omitted). Here is an explanation of what's going on.

Publications

A. Blum and M. Furst, "Fast Planning Through Planning Graph Analysis" [pdf], *Artificial Intelligence*, 90:281--300 (1997). This is the original paper that describes the algorithm used.

A. Blum and J. Langford, "Probabilistic Planning in the Graphplan Framework", in Proceedings of ECP'99. (c) Springer-Verlag. Describes how the planning graph structure can be used for probabilistic planning. See the Probabilistic Graphplan page.

Related Work

Since the initial creation of Graphplan, a number of researchers have pushed these ideas in a variety of exciting directions, including:

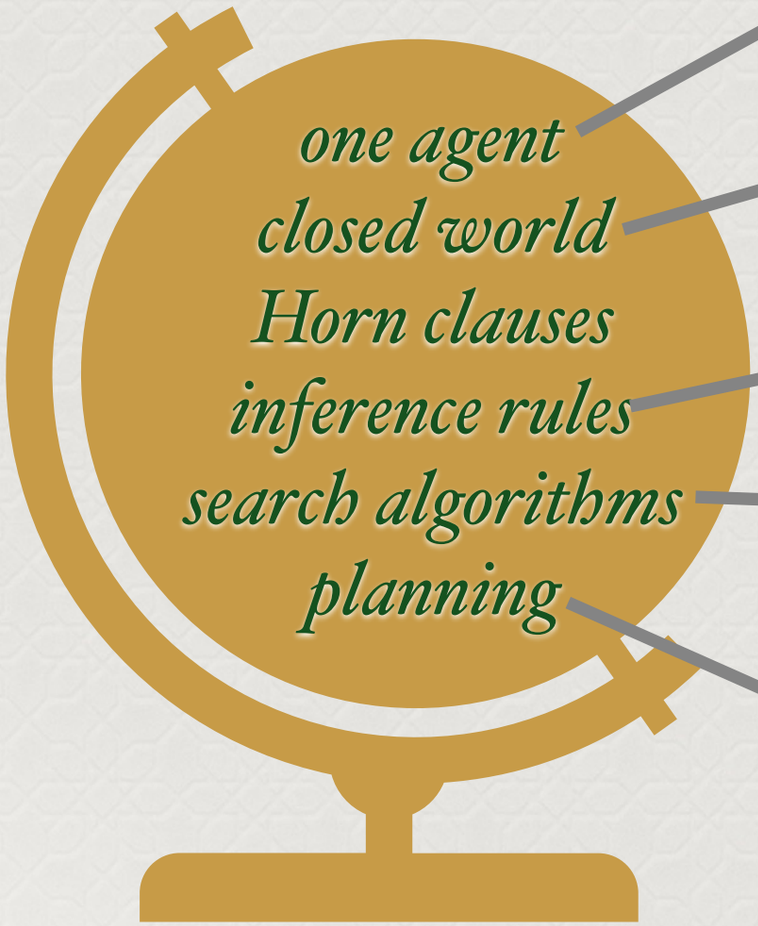
- broadening the class of problems for which this style of planning can be applied,
- reducing the running time (via improvements both to the basic strategy and to the code itself), and
- using Graphplan as a preprocessor to other search strategies.

In particular, check out the BLACKBOX (AT&T/Washington), IPP (U. Freiburg), STAN (U. Durham), and Sensory Graphplan (U. Washington) planners. See also John Langford's Plan compilation page.

Algorithms and Complexity | Computer Science Department | School of Computer Science

This page maintained by Avrim Blum (avrim@cs.cmu.edu). Last modified: June 2001

http://www.cs.cmu.edu/~avrim/graphplan.html



multiple agents

Kripke structures

dif. equation constraints

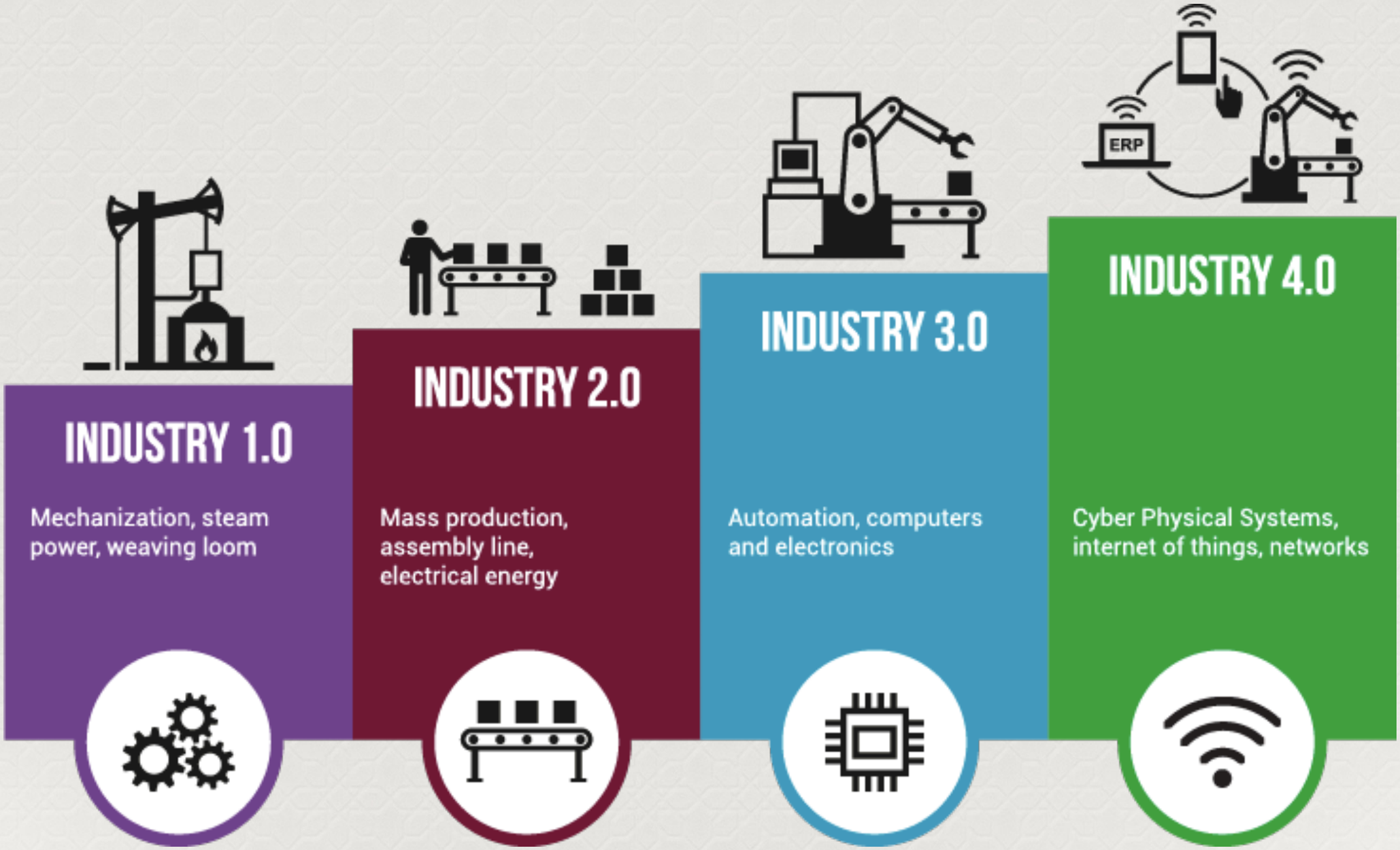
cognitive methods

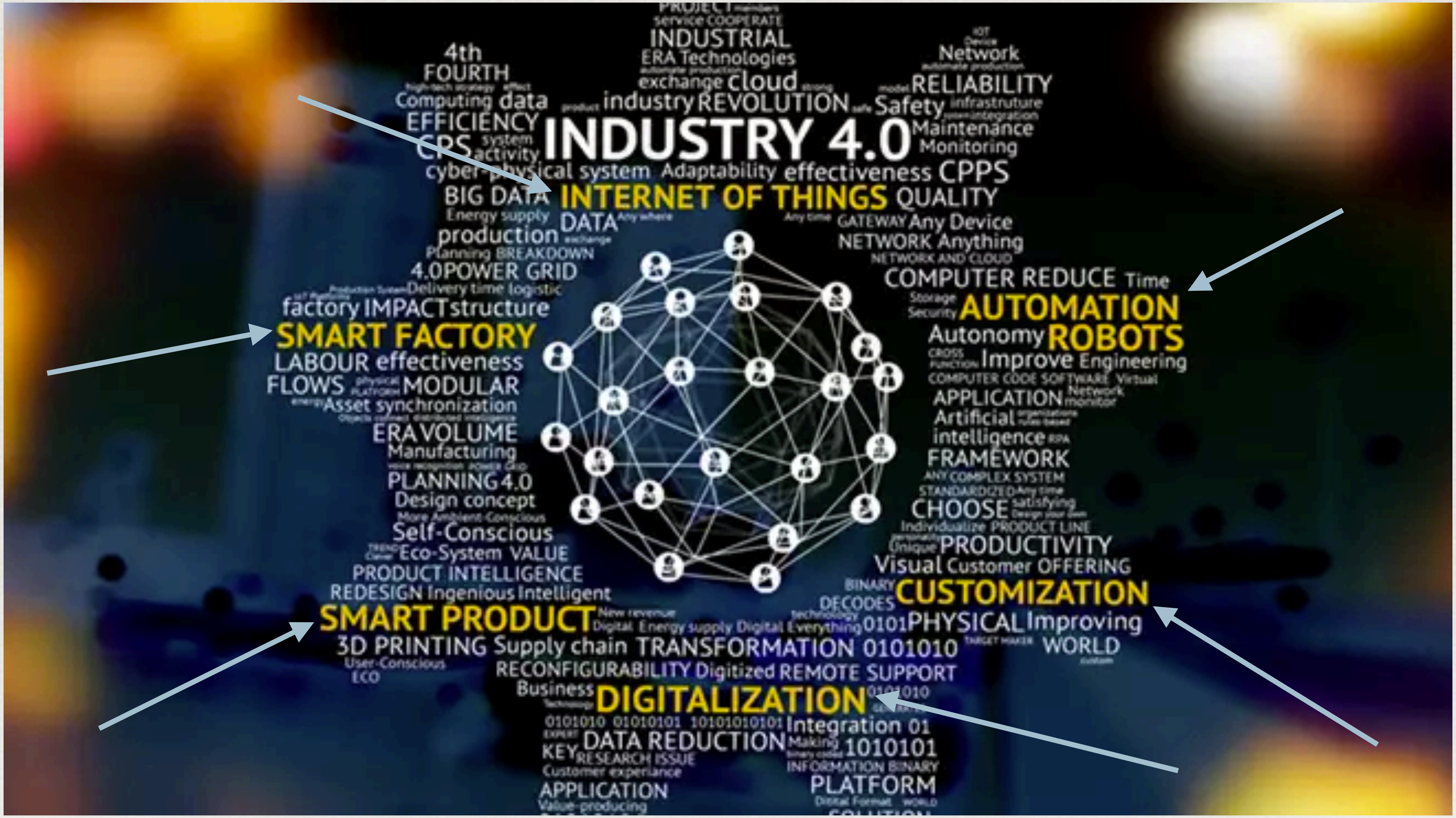
machine intelligence

A evolução da demanda das aplicações:

Industria 4.0

Orientação a serviços


ERP
INDUSTRY 1.0
Mechanization, steam power, weaving loom
INDUSTRY 2.0
Mass production, assembly line, electrical energy
INDUSTRY 3.0
Automation, computers and electronics
INDUSTRY 4.0
Cyber Physical Systems, internet of things, networks


INDUSTRY 4.0
INTERNET OF THINGS
SMART FACTORY
AUTOMATION
ROBOTS
SMART PRODUCT
CUSTOMIZATION
DIGITALIZATION
INDUSTRIAL
ERA Technologies
exchange cloud
industry REVOLUTION
4th
FOURTH
Computing data
EFFICIENCY
CPS
cyber-physical system
Adaptability effectiveness CPPS
BIG DATA
QUALITY
Network
RELIABILITY
Safety
Maintenance
Monitoring
production
DATA
GATEWAY Any Device
NETWORK Anything
COMPUTER REDUCE Time
Planning BREAKDOWN
4.0POWER GRID
Delivery time logistic
factory IMPACTstructure
LABOUR effectiveness
FLOWS MODULAR
Asset synchronization
ERA VOLUME
Manufacturing
PLANNING 4.0
Design concept
Self-Conscious
Eco-System VALUE
PRODUCT INTELLIGENCE
REDESIGN Ingenious Intelligent
Autonomy
Improve Engineering
COMPUTER CODE SOFTWARE Virtual
APPLICATION
Artificial
intelligence
FRAMEWORK
CHOOSE
PRODUCTIVITY
Visual Customer OFFERING
PHYSICAL Improving
WORLD
3D PRINTING Supply chain TRANSFORMATION
RECONFIGURABILITY Digitized REMOTE SUPPORT
Business
Integration
DATA REDUCTION
KEY
RESEARCH ISSUE
Customer experiance
APPLICATION
INFORMATION BINARY
PLATFORM

Outro aspecto de grande importância e atualidade é que a base do sistema produtivo está mudando de um foco baseado em ítens (produtos) para serviços.

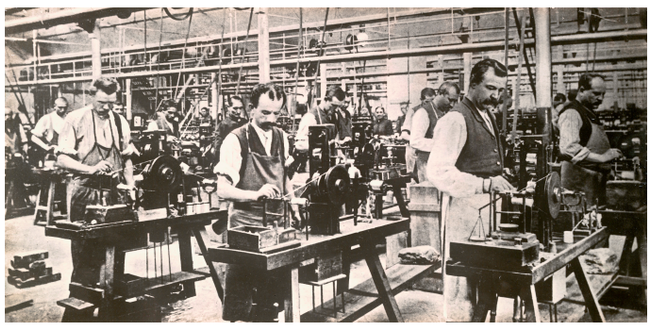
70%

20%

10%

50%

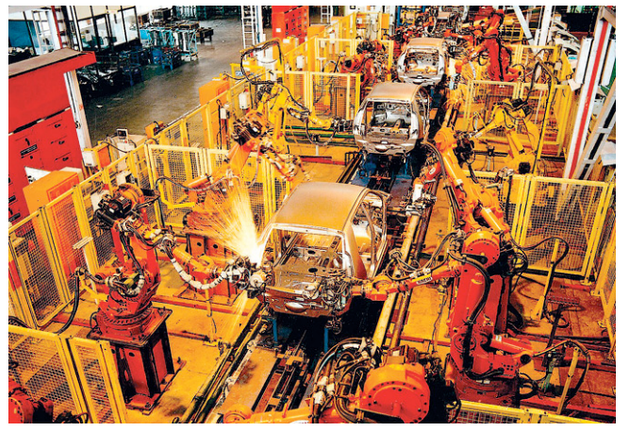
30%



20%

5%

25%



70%

primary

secondary

tertiary

t

A “ciência de serviços"

A ciência de serviços é ciência das interações produtor-consumidor, onde o produtor deve satisfazer as necessidades individuais do consumidor extraídas do seu perfil, propriedades, e de informação provida por este mesmo consumidor.

Sampson, S., 2007. A customer-supplier paradigm for service science. In: DSI Service Science Miniconference. No. February. Pittsburgh, pp. 11–16. URL http://sampson.byu.edu/dsimini/proc/docs/1-3822.pdf

A ciência de serviços é o estudo da composição formal de unidades de transformação e criação de valor em unidades maiores e mais complexas. Pode assim ser aplicada à produção de bens tangíveis ou intangíveis, onde os elementos de composição são modelos completos.

Silva, J. R., 2014. D-Lab, Escola Politécnica da USP

Industry new era

Service Science

Good-dominant

vs

Service-dominant

"Servitization"

Industry 4.0

Embedded Systems

Networked
Embedded Systems

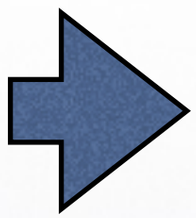
Cyber-Physical Systems

Industry new era

Service Science

Industry 4.0

Product-Service Systems



Smart products (Cyber-physical)

Co-production structure

PSA

Smart Factories

PSA - Product-Service Architecture

Manufacturing Services:

- Open Set of value-production unities
- Connected or independent of a corporation

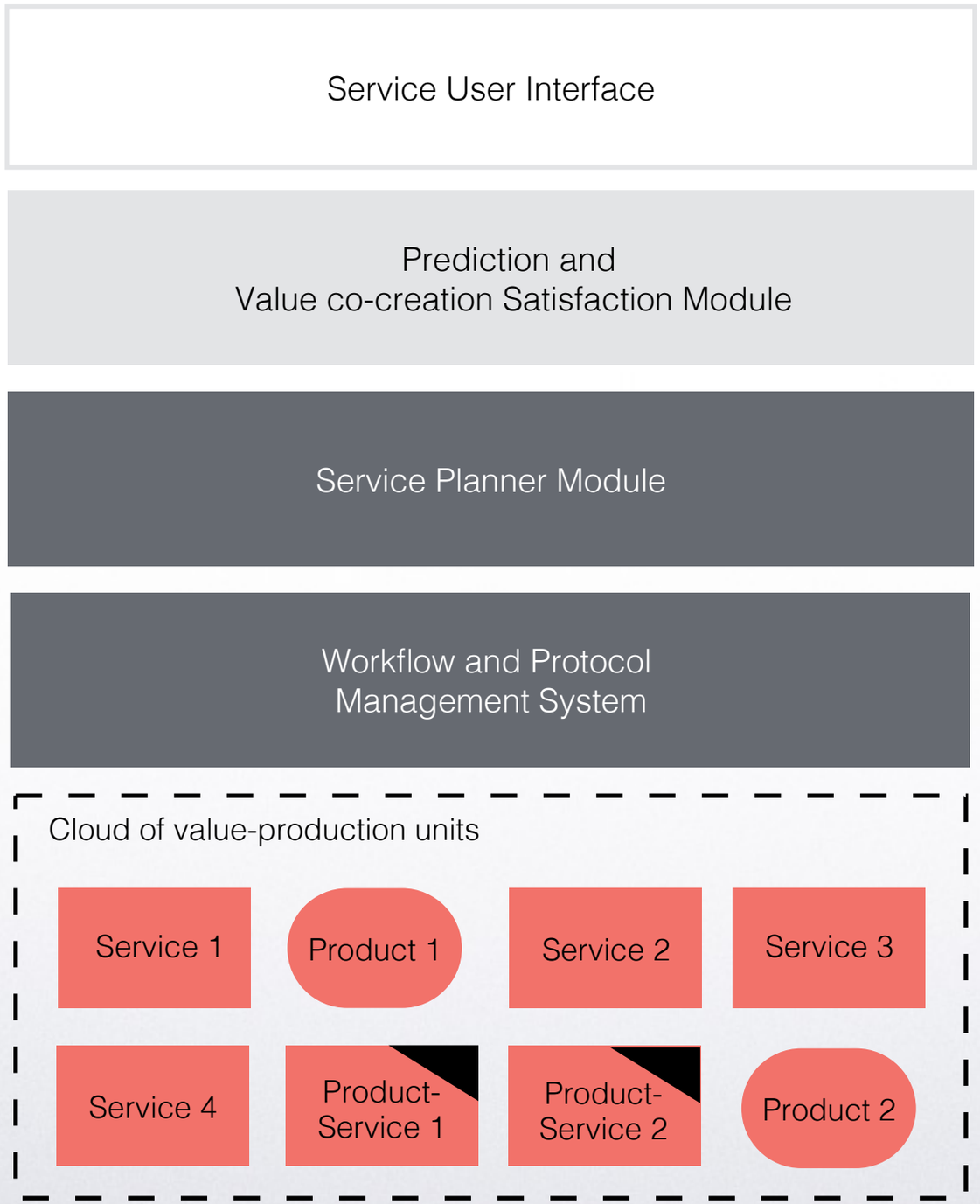
Value production arrangement:

- Provide a product/service coupling with a final costumer
- Workflow, Plan, Coordination of the set of VPU

(Silva and Nof, 2015)

7

PSA

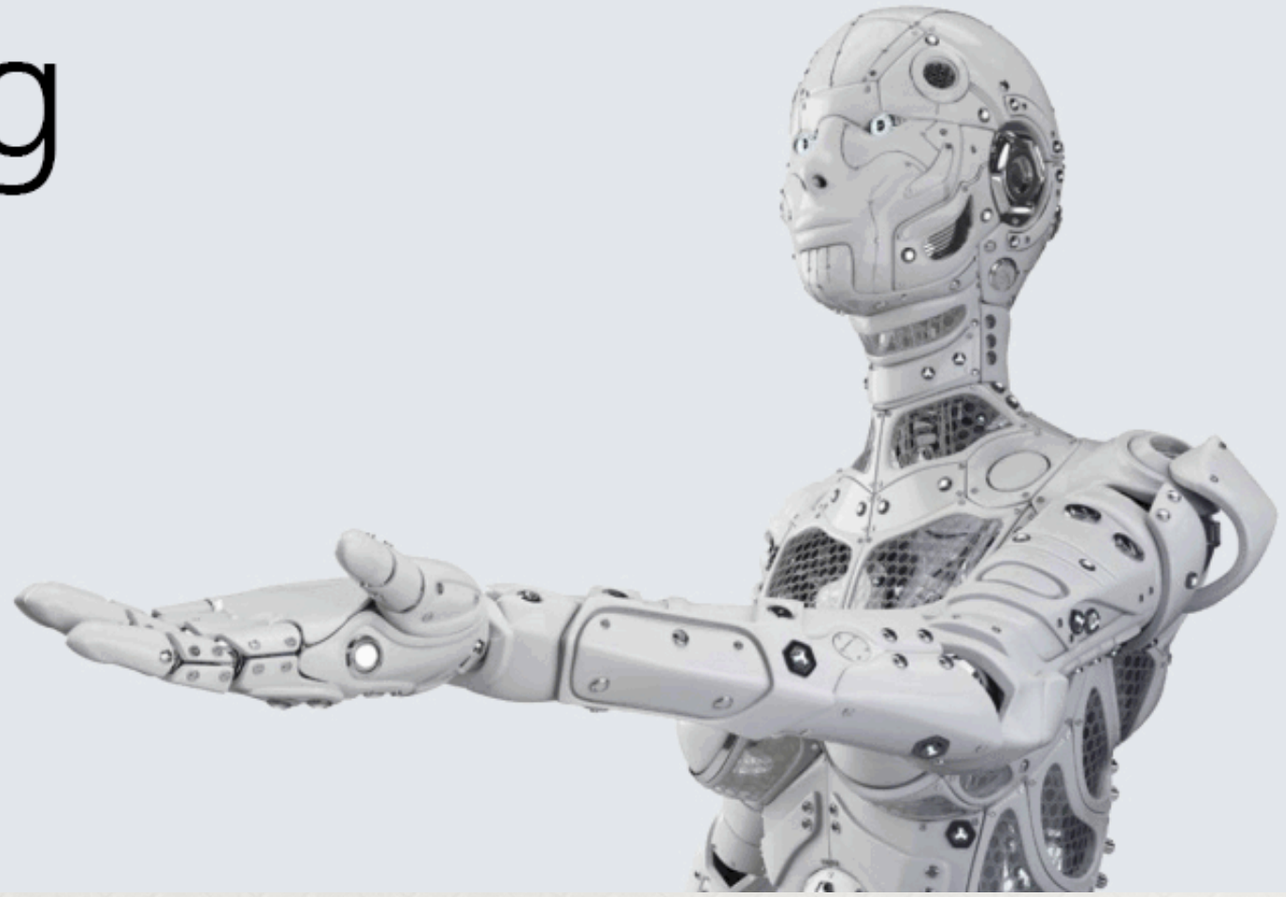


Collaborative e-work Systems

(Silva and Nof, 2015)

AI's coming of age

The progress into the AGI phase and the beginning of true autonomy.



Chegamos ao final!