Automated Planning-as-a-Service

Theory and Practice

Andrea Marrella

Associate Professor – Sapienza University of Rome

Outline

- Introduction to Automated Planning in AI
- Automated Planning as-a-Service
- Context-Aware Trace Alignment via Automated Planning
- Concluding Remarks

Outline

- Introduction to Automated Planning in AI
- Automated Planning as-a-Service
- Context-Aware Trace Alignment via Automated Planning
- Concluding Remarks

AI AND AUTONOMOUS BEHAVIOUR

At the center of the problem of autonomous behavior is the **control problem** (or **action selection problem**).

- *specify a **controller** that selects the action to do next*

Hard-coded solutions specify a **pre-scripted controller** in a high-level language.

- ✔ They do not suffer combinatorial explosion.
- ➖ The burden is all put on the programmer.
- ➖ Hard-coded solutions tend to constraint the search in some way.

The question of **action selection** for AI researchers is:

- ***What is the best way to intelligently constrain this search?***

DEALING WITH AUTONOMOUS BEHAVIOR IN AI

Learning-based approach

- The controller is **learnt from experience**.
  - ✔ Discovery and interpretation of meaningful patterns for a given task.
  - ⛔ Learned solutions are usually black-box.

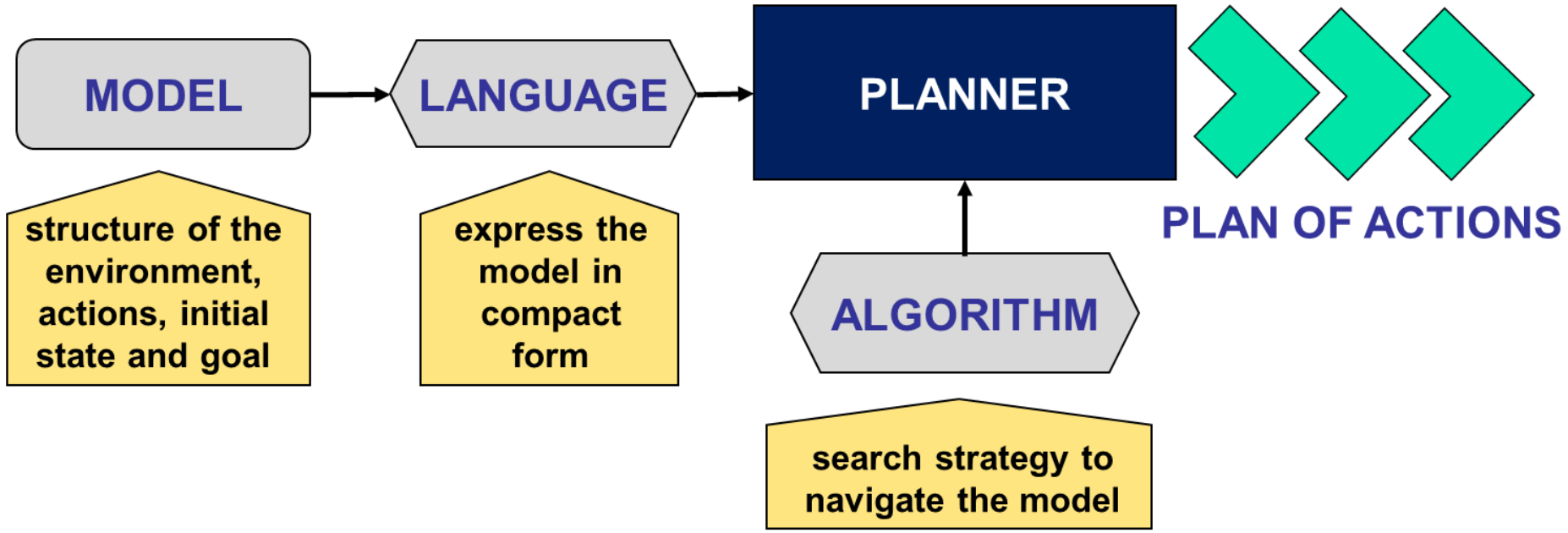
Model-based approach

- The controller is **derived automatically** from a model of the domain of interest, the actions, the current state, and the goal.
  - ✔ The models are all conceived to be general.
  - ⛔ The problem of solving a model is computationally intractable.

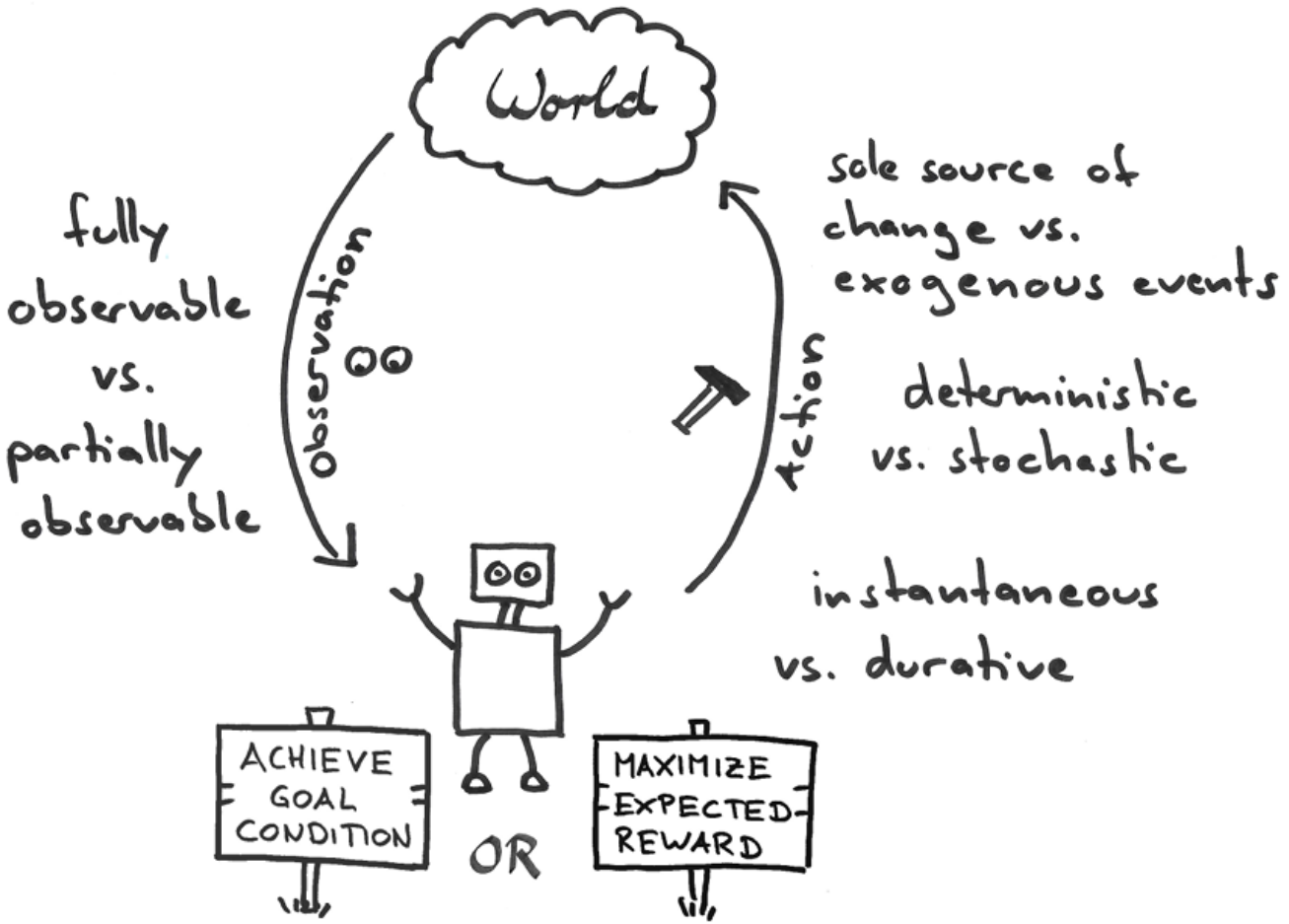
Focus on **Automated Planning**

AUTOMATED PLANNING IN AI [Geffner-M&C13]

- **Automated planning** is a *model-based approach for the automated synthesis of plans of actions to achieve goals.*

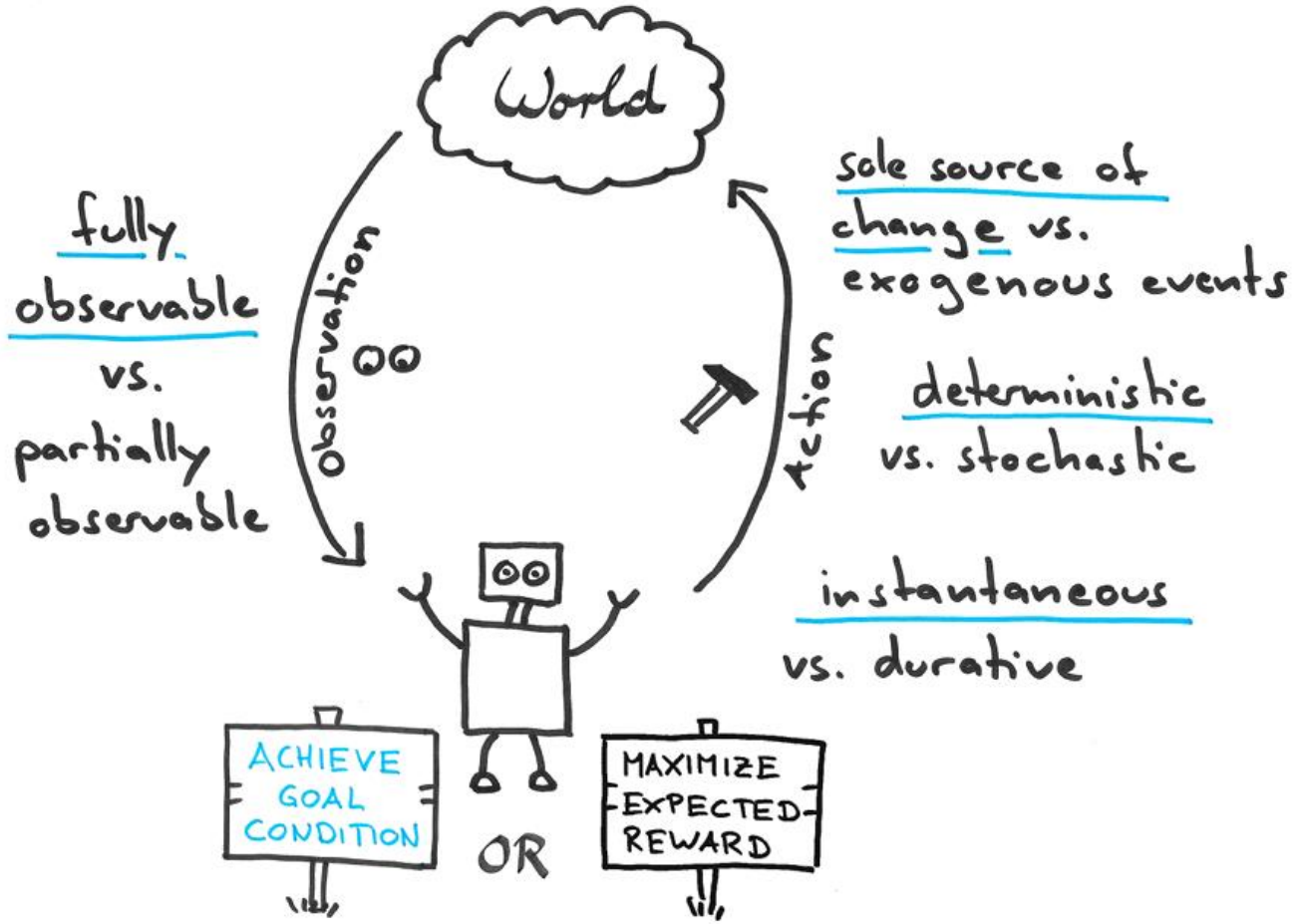


PLANNING MODELS



Credits to M.Helmert, G.Roger (University of Basel)

CLASSICAL PLANNING MODEL



Credits to M.Helmert, G.Roger (University of Basel)

CLASSICAL PLANNING MODEL

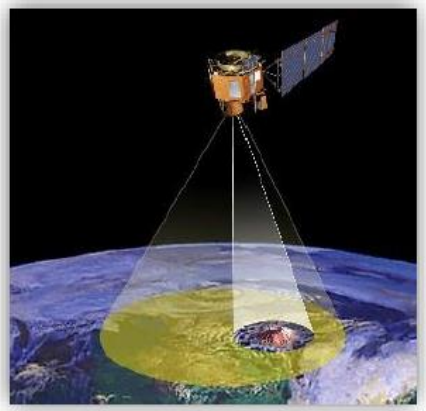
- **finite** and **discrete** state space S
- a **known initial state** $I \in S$
- a set $S_G \subseteq S$ of **goal states**
- **actions** $A(s) \subseteq A$ applicable in each $s \in S$
- a **deterministic transition function** $s' = f(a, s)$ for $a \in A(s)$
- positive **action costs** *c(a,s)*

- A **solution** or **plan** is a sequence of applicable actions $\pi = a_0, \ldots, a_n$ that maps I into S_G
  - There are states $s_0, \ldots, s_{n+1}$ such that $s_{i+1} = f(a_i, s_i)$ and $a_i \in A(s_i)$ for $i = 0, \ldots, n$ and $s_{n+1} \in S_G$
- A plan is **optimal** if it minimizes the sum of action costs $\sum_{i=0,\ldots,n} c(a_i, s_i)$.
  - If costs are all 1, plan cost is plan length.

COMPLEXITY OF CLASSICAL PLANNING

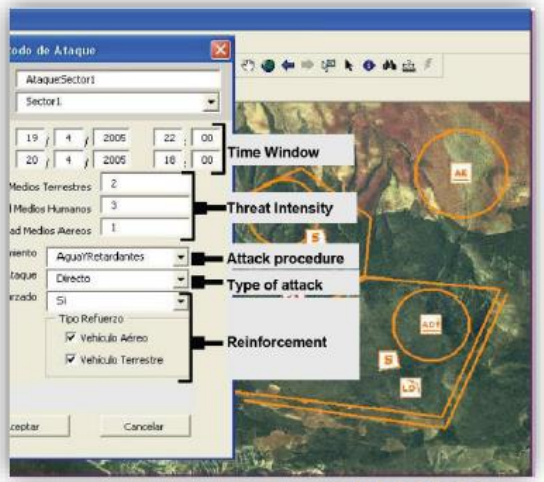
- Planning: thinking before acting
- Classical Planning: the “easy” case (deterministic, fully observable, etc.)
- Solved by **state-space search**
- Computationally **intractable** even for the simplest models.
  - **PSPACE-complete** because of huge number of states.
  - Number of states grow exponentially with the input size
- Classical planners **solve efficiently** real problems with hundreds of propositions!

EXAMPLES OF PLANNING PROBLEMS: EARTH AND SPACE

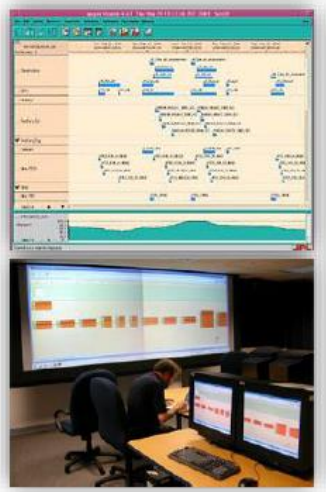


Satellites can not always communicate with ground operators.

On-board planning
to view interesting natural events:
http://ase.jpl.nasa.gov/



SIADEX –
plan for firefighting
Limited resources
Plan execution is dangerous!



NASA Mapgen / Mars Rovers

Primary platform for creating daily activity plans for Spirit, Opportunity

Mixed-initiative tool:
Human in the loop

Robots on other planets may be hours away by radio.

AUTOMATED PLANNING IN SOC AND BPM FIELDS

ELSEVIER

Artificial Intelligence

Volume 236, July 2016, Pages 30-64

Domain-independent planning for services in uncertain and dynamic environments

Eirini Kaldeli [a b] ✉, Alexander Lazovik [b], Marco Aiello [b]

Automated Planning for Ubiquitous Computing

Authors: Ilche Georgievski and Marco Aiello | Authors Info & Claims

ACM Computing Surveys (CSUR), Volume 49, Issue 4 • Article No.: 63, Pages 1 - 46 • https://doi.org/10.1145/3004294

Published: 05 December 2016 Publication History

Check for updates

AUTOMATED PLANNING IN SOC AND BPM FIELDS

Home > Journal on Data Semantics > Article

Automated Planning for Business Process Management

Original Article | Published: 01 November 2018

Volume 8, pages 79–98, (2019) Cite this article

Download PDF

Access provided by CARE-CRUI COMPACT

Journal on Data Semantics

Andrea Marrella

Use our pre-submission checklist →

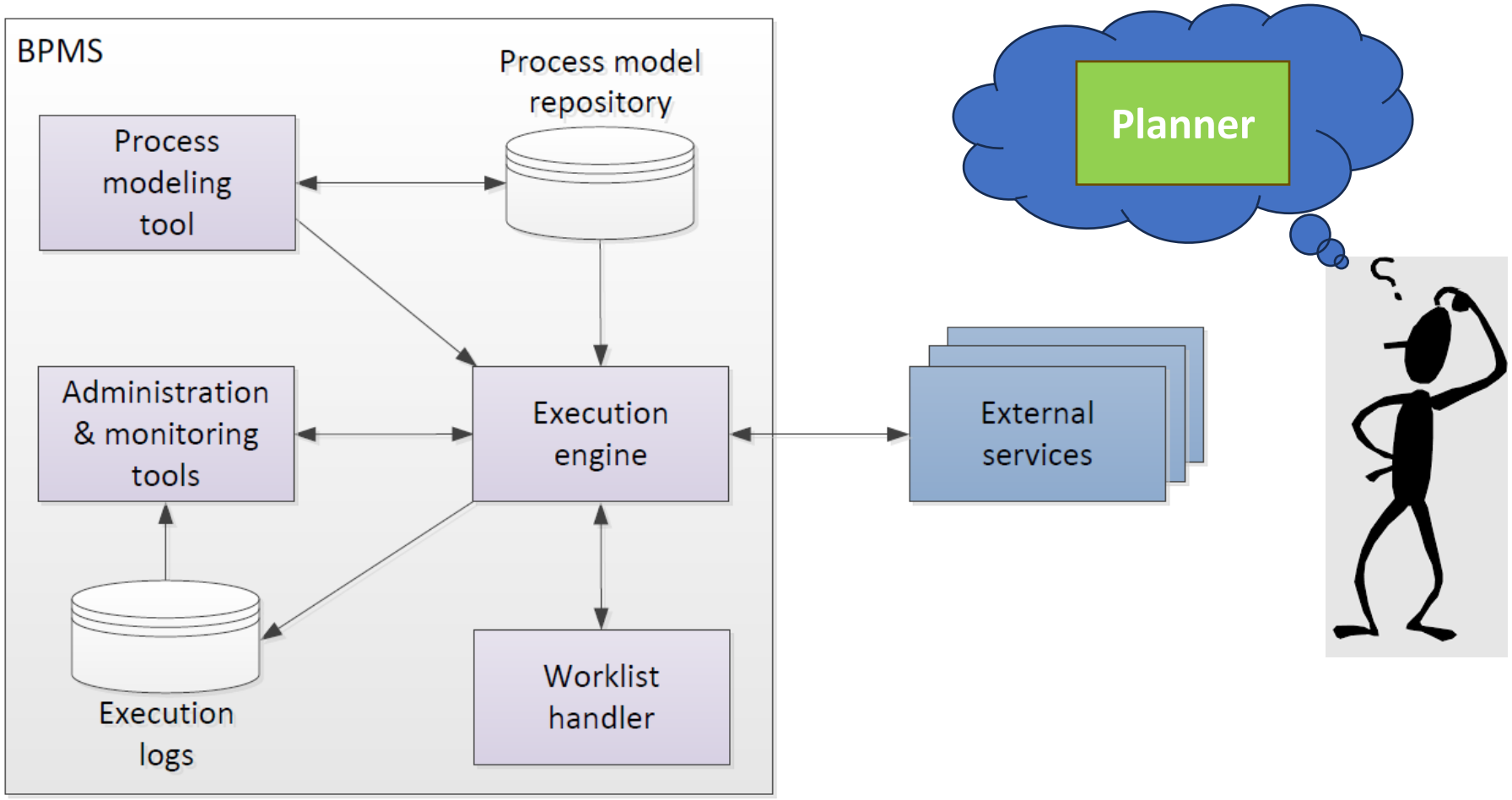
Outline

- Introduction to Automated Planning in AI
- Automated Planning as-a-Service
- Context-Aware Trace Alignment via Automated Planning
- Concluding Remarks

HOW TO INVOKE A PLANNER AS-A-SERVICE?

Software systems can invoke planners as **external services**.

No expertise of the internal working of the planners is required to build a plan.



Credits to M. Dumas, et al. Fundamentals of BPM. Springer 2018

PDDL - PLANNING DOMAIN DEFINITION LANGUAGE

Basic components of a PDDL planning task:

- **Objects**: Things in the world that interest us.
- **Predicates**: Properties of objects that we are interested in; they can be *true* or *false*.
- **Initial state**: The state of the world that we start in.
- **Goal specification**: Things that we want to be true.
- **Actions/Operators**: Ways of changing the state of the world.

Planning tasks specified in PDDL are separated into two parts/files:

- A ***planning domain*** file for predicates and actions.
- A ***planning problem*** file for objects, initial state and goal specification.
    - The domain models the relevant aspects of the world in which we are planning, while the problem definition is a specific problem instance in the domain that specifies ***where we begin*** and ***what we must achieve***.

HISTORY AND FRAGMENTS OF PDDL

[1998] PDDL 1.2

Basic version with **STRIPS**, **ADL** and **conditional effects**.

[2002] PDDL 2.1

It introduces **numeric fluents** (e.g., to model *non-binary resources* such as distance, weight, temporal features, etc.), **plan-metrics** (to allow *quantitative evaluation* of plans, and not just goal-driven), and **durative/continuous actions** (which could have variable, non-discrete length, conditions and effects).

[2004] PDDL 2.2

It introduces **derived predicates** (to model the dependency of given facts from other facts), and **timed initial literals** (to model exogenous events occurring independently from plan-execution).

[2006] PDDL 3.0

It introduces **preferences** (hard- and soft-constraints, in form of logical expressions, to be satisfied in specific points of the plan).

[2008] PDDL 3.1

It introduces **object fluents** (functions' range can be any object-type).

DOMAIN FILE

PDDL skeleton of a planning domain file:

```
(define (domain <domain name>)
  (:requirements :strips)
  (:predicates
    <PDDL code for predicates>
  )
    <PDDL code for first action>
    [...]
    <PDDL code for last action>
)
```

<domain name> identifies the name of the planning domain, e.g., blocks-world.

PROBLEM FILE

PDDL skeleton of a planning problem file:

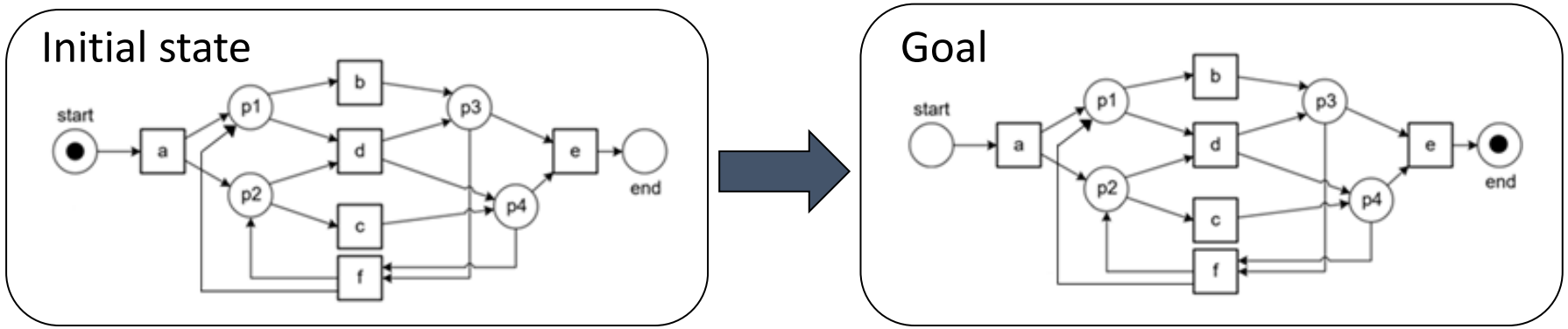
```
(define (problem <problem name>)
 (:domain <domain name>)
  <PDDL code for objects>
  <PDDL code for initial state>
  <PDDL code for goal specification>
)
```

`<problem name>` identifies the name of the specific planning task, e.g., `blocks1`.

`<domain name>` matches the domain name in the corresponding domain file, e.g., `blocks-world`.

EXAMPLE: REACHABILITY IN PETRI NETS

Given a Petri Net (PN) with an initial marking m_0 and a target marking m_n, if there exists a sequence of transition firings $<t_1 \ldots t_n>$ that leads from m_0 to m_n, then m_n is said to be **reachable** from m_0



One available action: **firing**

- If t is *enabled*, firing t changes the marking of the PN.
  - Each token from the input places of t is consumed, and one token is produced in any output place of *t*

THE REACHABILITY PROBLEM IN PDDL

Planning Domain

```
(define (domain petri-net)
   (:types place transition)
   (:predicates (token ?p - place)
                (input_place ?t - transition ?p - place)
                (output_place ?t - transition ?p - place))
   (:action fire
       :parameters (?t - transition)
       :precondition (forall (?p - place)
                        (imply (input_place ?t ?p)
                               (token ?p)))
       :effect (and (forall (?p - place)
                       (when (input_place ?t ?p)
                             (not (token ?p))))
                    (forall (?p - place)
                       (when (output_place ?t ?p)
                             (token ?p)))
                    (increase (total-cost) 1)))
)
```

Objects of the domain and **predicates** describe the **state** of the world.

Actions are described in terms of **preconditions** under which an action can be executed, and **effects** on the state of the world, stated in terms of the predicates.

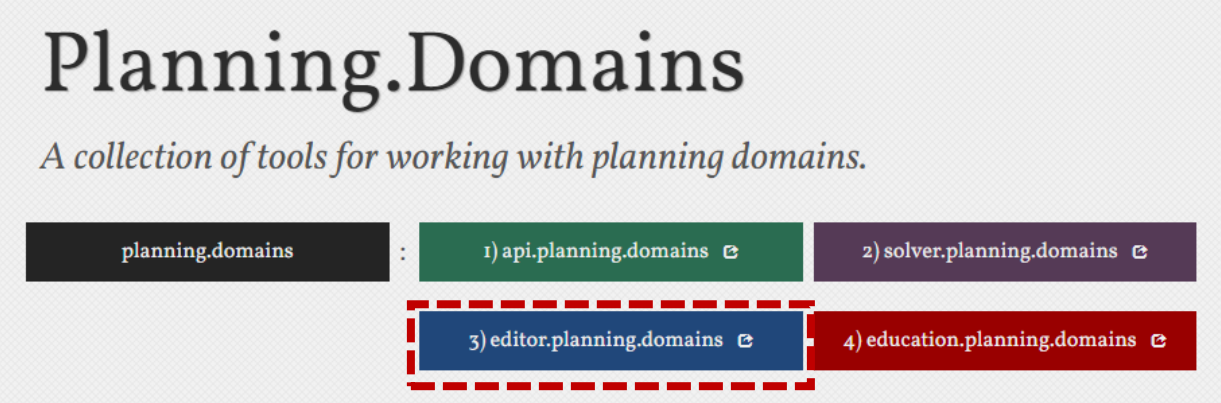
THE REACHABILITY PROBLEM IN PDDL

Planning Problem

```
(define (problem pr1)
   (:domain petri-net)
   (:objects start p1 p2 p3 p4 end - place
                  a b c d e f - transition)
   (:init (token start) (input_place a start)
          (output_place a p1) (output_place a p2)
          (input_place b p1) (output_place b p3)
          (input_place c p2) (output_place c p4)
          (input_place d p1) (input_place d p2)
          (output_place d p3) (output_place d p4)
          (input_place e p3) (input_place e p4)
          (output_place e end) (input_place f p3)
          (input_place f p4) (output_place f p2)
          (= (total-cost) 0))
   (:goal (and (token end) (not (token start))
               (not (token p1)) (not (token p2))
               (not (token p3)) (not (token p4))))
   (:metric minimize (total-cost))
)
```

SPECIFYING A PDDL MODEL

- Many tools are valuable in the process of writing (and debugging) a PDDL model.
- For simple problems, a good solution is http://planning.domains, an on-line repository of planning benchmark models that includes:
  - an ***on-line editor*** with PDDL-specific features such as syntax highlighting and semi-automatic instantiation of some common model patterns.
  - an interface to a ***planner as a web service***. However, the planner it makes available does not support all PDDL, and as it has a runtime limit of 15 seconds it is restricted to solving small problems.



PRODUCING A PLAN

- State-of-the art planners* provide **customized implementations** of the search algorithms with varying features of completeness, optimality, and memory complexity.

> **Cf. http://icaps-conference.org/index.php/main/competitions

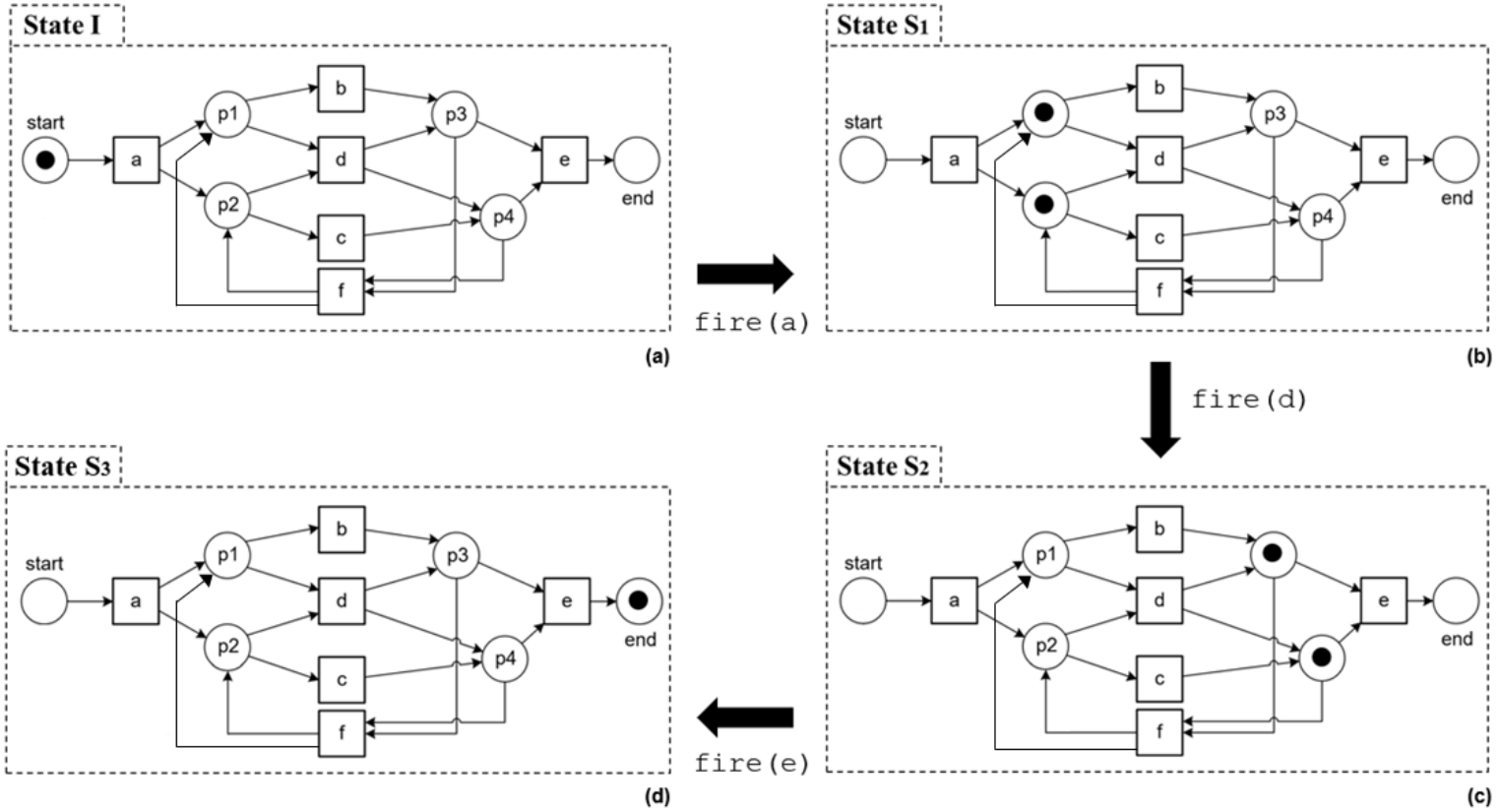
- **Generality**: A planner can solve ***arbitrary problem instances***.
  - A planner does not know what the actions, and domain stand for.
  - This is very different from writing a *domain-specific* solver.

- **Scalability**: Planners embed very effective ***domain-independent heuristics*** to drive the searching task towards the goal.
  - A ***heuristic function*** provides an estimate of the cost to reach the goal from the current state (Examples: Best-First Search, A*, Hill Climbing, etc).

THE REACHABILITY PROBLEM IN PDDL

Optimal plan

```
Begin plan
1. (fire a)
2. (fire d)
3. (fire e)
End plan
```



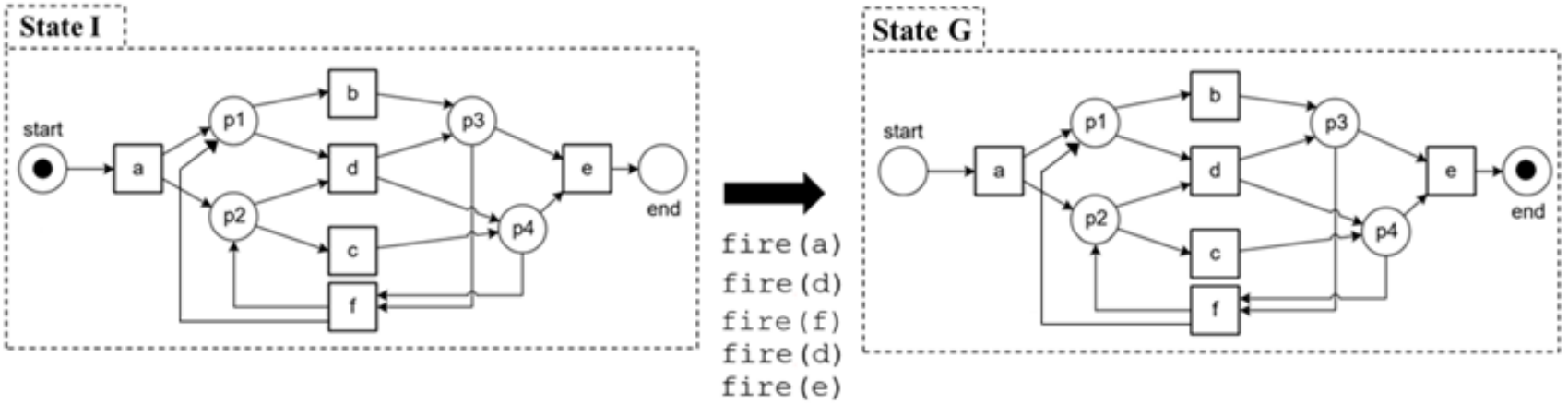
Since S_3 is a state satisfying G, the solution found is a **valid plan**.

THE REACHABILITY PROBLEM IN PDDL

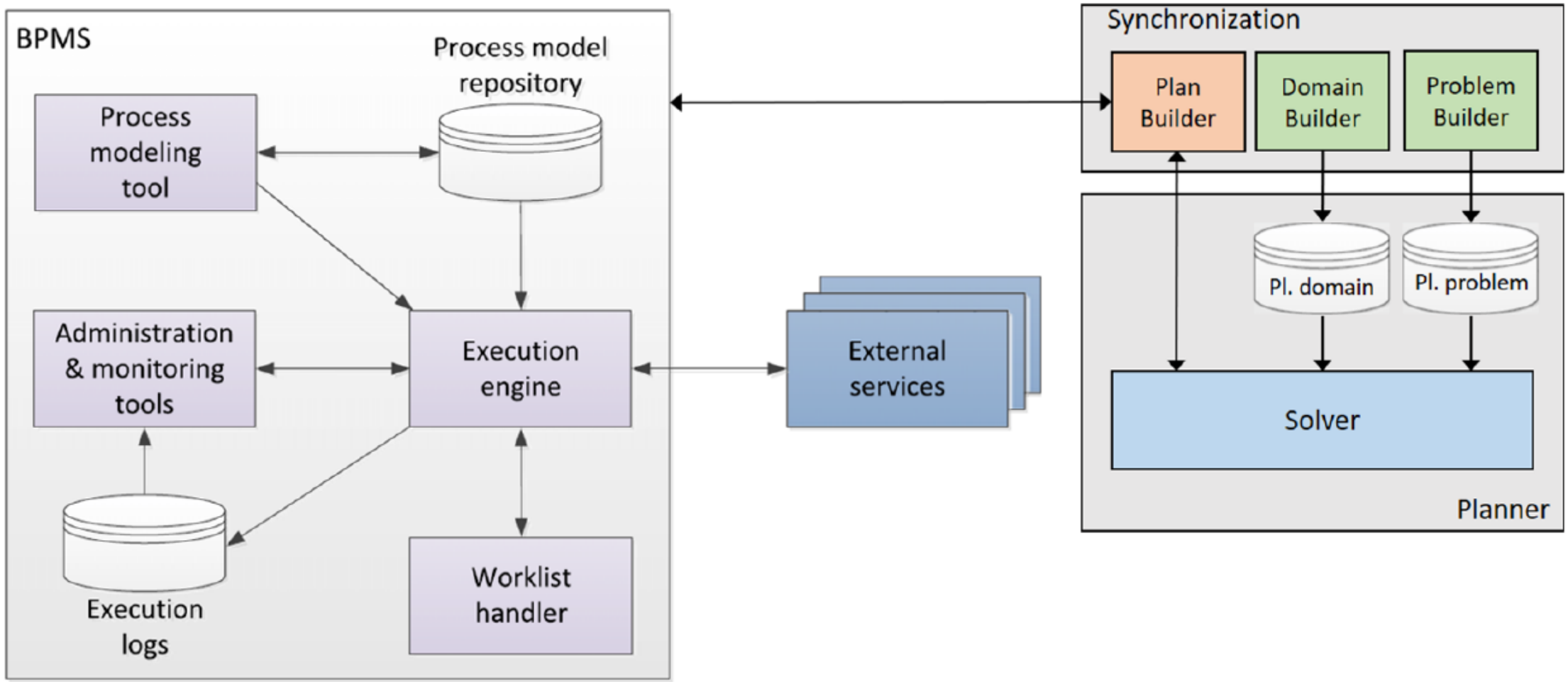
Suboptimal plan

The **quality of a solution** depends by the specific search algorithm employed by the planner.

```
Begin plan
1. (fire a)
2. (fire d)
3. (fire f)
4. (fire d)
5. (fire e)
End plan
```



INVOKING A PLANNER AS-A-SERVICE

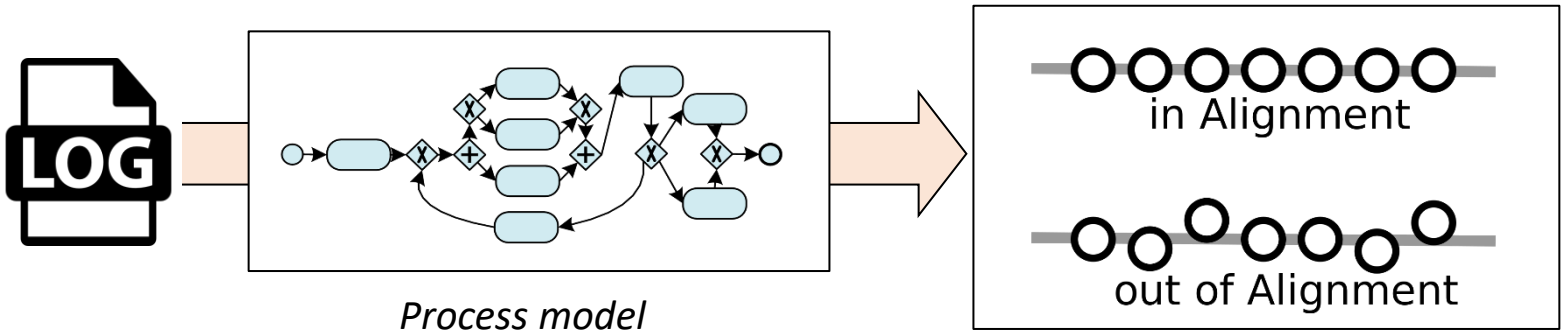


Outline

- Introduction to Automated Planning in AI
- Automated Planning as-a-Service
- Context-Aware Trace Alignment via Automated Planning
- Concluding Remarks

TRACE ALIGNMENT

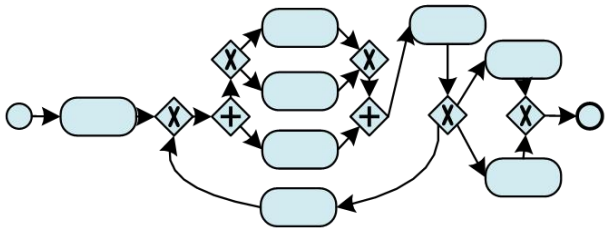
- **Process models** are typically not fully enforced by information systems (human behavior is often involved).
  - Traces of execution can be **dirty** with **spurious** or **missing events**.
  - Possible **discrepancies** between the modeled and the observed behavior.



Process model

- **Trace Alignment** finds the **best execution sequence** of a process model (**optimal alignment**) that reproduces an execution trace of the process by pinpointing where it deviates [Carmona-van-Dongen-18].

EVENT LOGS



Process model

Information system

Event log

XES
Extensible Event Stream

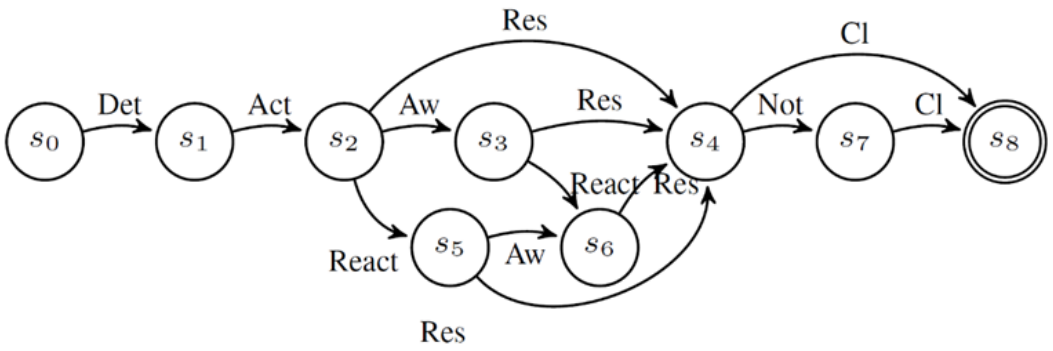
www.xes-standard.org

case id	event id	properties				
		timestamp	activity	resource	cost	...
1	35654423	30-12-2010:11.02	register request	Pete	50	...
	35654424	31-12-2010:10.06	examine thoroughly	Sue	400	...
	35654425	05-01-2011:15.12	check ticket	Mike	100	...
	35654426	06-01-2011:11.18	decide	Sara	200	...
	35654427	07-01-2011:14.24	reject request	Pete	200	...
2	35654483	30-12-2010:11.32	register request	Mike	50	...
	35654485	30-12-2010:12.12	check ticket	Mike	100	...
	35654487	30-12-2010:14.16	examine casually	Pete	400	...
	35654488	05-01-2011:11.22	decide	Sara	200	...
	35654489	08-01-2011:12.05	pay compensation	Ellen	200	...
3	35654521	30-12-2010:14.32	register request	Pete	50	...
	35654522	30-12-2010:15.06	examine casually	Mike	400	...
	35654524	30-12-2010:16.34	check ticket	Ellen	100	...
	35654525	06-01-2011:09.18	decide	Sara	200	...
	35654526	06-01-2011:12.18	reinitiate request	Sara	200	...
	35654527	06-01-2011:13.06	examine thoroughly	Sean	400	...
	35654530	08-01-2011:11.43	check ticket	Pete	100	...
	35654531	09-01-2011:09.55	decide	Sara	200	...
	35654533	15-01-2011:10.45	pay compensation	Ellen	200	...

- Each **event** in a log refers to an activity of the process and is included in a **trace** (case), i.e., a run of the process.
- Each event is associated with:
  - A trace/case id
  - A timestamp
  - A process activity
  - (optional) a list of attributes

ALIGNMENT WITH STATIC COST FUNCTION

- **Limiting assumption**: state-of-the-art alignment algorithms are driven by a **static cost function** assigning fixed costs to deviations.
- The **context** in which a deviation is found is **neglected**.



ISO/IEC 27035 – Incident Management process

Det – Detect an incident
Act – Register a ticket
Res – Incident resolution
Aw – Assessment made by 3rd party companies
React – Reactivate the ticket
Not – Resolution notification
Cl – Ticket closure

t_1 = ⟨Act, Aw, Aw, Aw, Res, Cl⟩
t_2 = ⟨Act, Act, Act, Aw, Res, Cl⟩

Alignment

t_1 = ⟨**Det**, Act, Aw, ~~**Aw**~~, ~~**Aw**~~, Res, Cl⟩
t_2 = ⟨**Det**, Act, ~~**Act**~~, ~~**Act**~~, Aw, Res, Cl⟩

Unitary cost

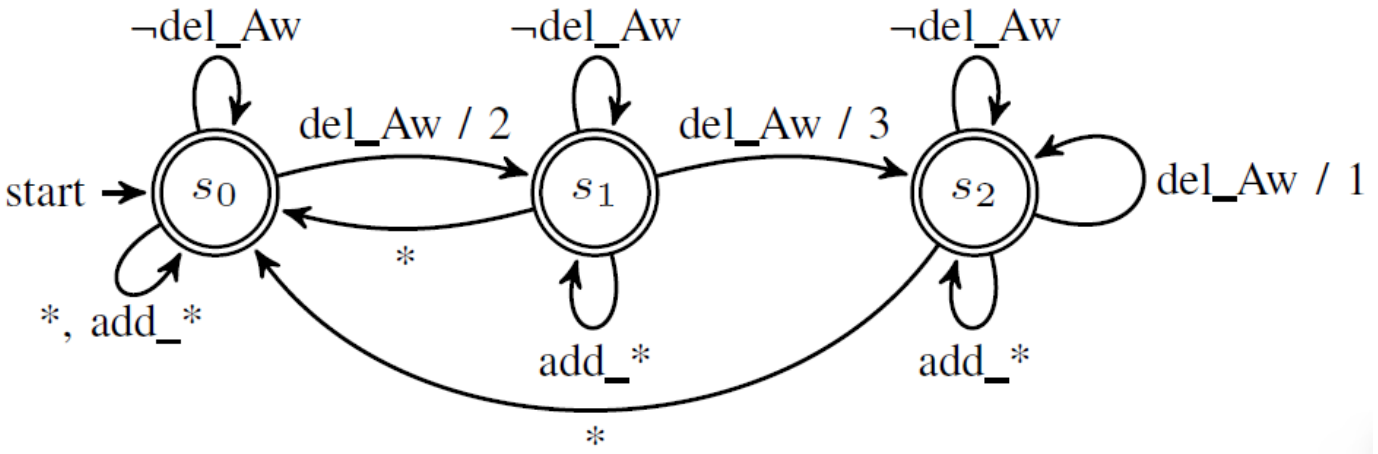
Cost(t_1) = 3
Cost(t_2) = 3

Repeated **Aw** → process execution is stuck.
Repeated **Act** → no impact for the security team.

Are equally problematic?
The cost of repeating AW should increase at any occurrence!

SEVERITY BASED ON COST MODEL

- **Severity** of a deviation → based on the **context** in which it is found.
- **Context**: *determined by the position in the trace in which the deviation occurs and, more specifically, by the activities or sequence of activities that occur before or after the deviation*.
- Notion of "context" expressed through a dedicated ***cost model***
  - **Context-dependent variable costs** associated to the deviations that may occur in an alignment.



TRACE ALIGNMENT AS A THEORETIC MANIPULATION OF DFAs

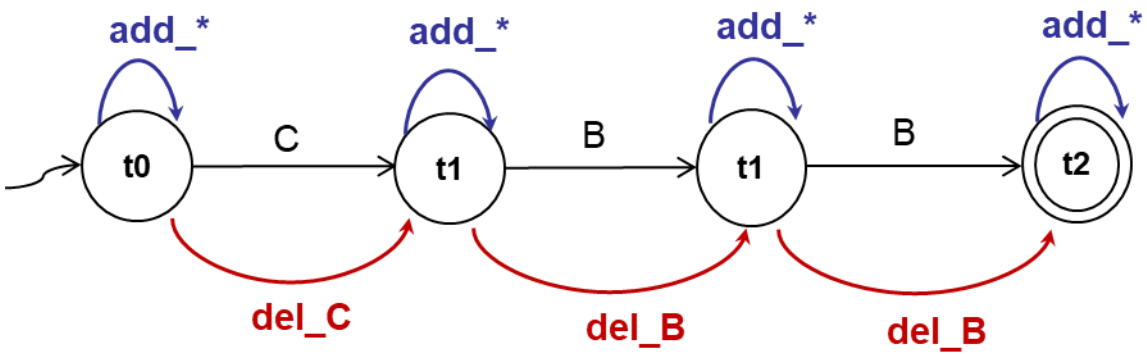
Target: Generate optimal alignments driven by cost models.

Approach:

1. Process models as **Deterministic Finite State Automata** (**DFAs**)
   - **Clear semantics** to perform formal reasoning over the process model.
   - **Not directly tied** to the prescriptive/declarative nature of the process.
2. DFA-theoretic manipulations to specify the **alignment instructions**.
3. Recasting as a **cost-optimal planning problem** in AI.
4. Automated **planning technology** to find **optimal alignments**.

DFA-BASED SOLUTION /1

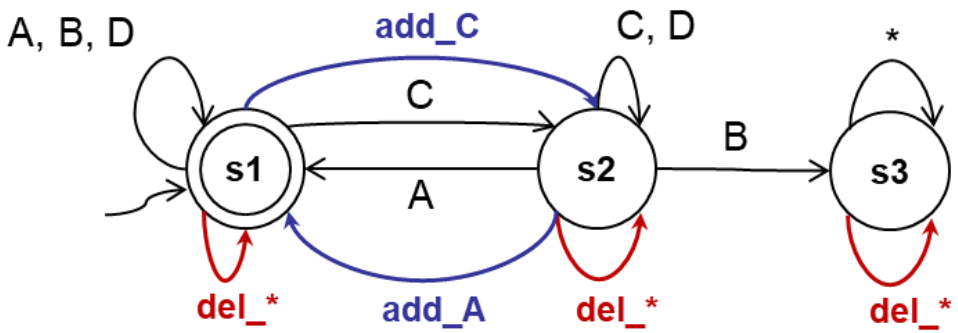
- Context-aware trace alignment can be solved using DFAs [Acitelli-ICPM22].
  - One augmented DFA for the partial trace.



  - Accepts input trace (<C,B,B>) plus all other traces, however…
  - …changes wrt. input trace must be marked by add/del, e.g.,
    - <A,B,C> = del_C add_A B del_B add_C

DFA-BASED SOLUTION /2

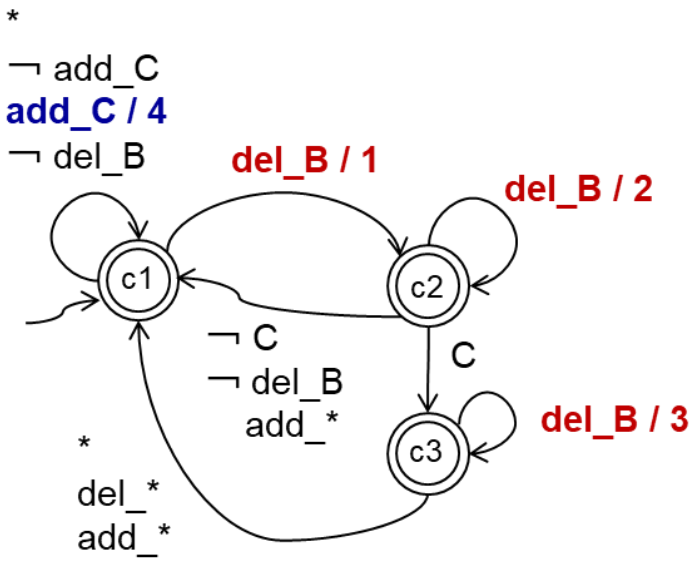
- DFA representing the process model augmented to account for **add** and **del** activities.



- Accepts all (possibly repaired) traces satisfying the model.

DFA-BASED SOLUTION /3

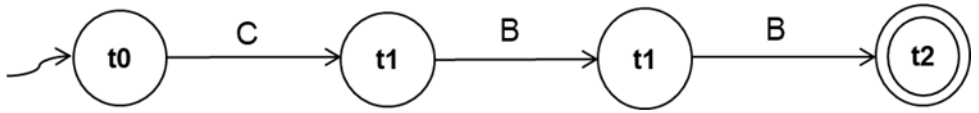
- One (or more) DFA(s) to represent the cost model.
- The states of the cost model are all final.
- When **add**/**del** activities are not related to any cost, we associate them with the fixed cost captured by the static cost function.



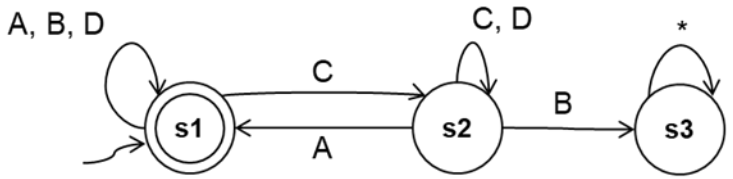
Context-aware trace alignment *is the problem of finding a sequence of synchronous steps performed in all augmented DFAs and in the cost model such that -- at the end of the alignment -- each DFA is in at least one accepting state and the* ***cost of the alignment is minimal****.*

EXAMPLE OF CONTEXT-AWARE TRACE ALIGNMENT

Trace: <C,B,B>

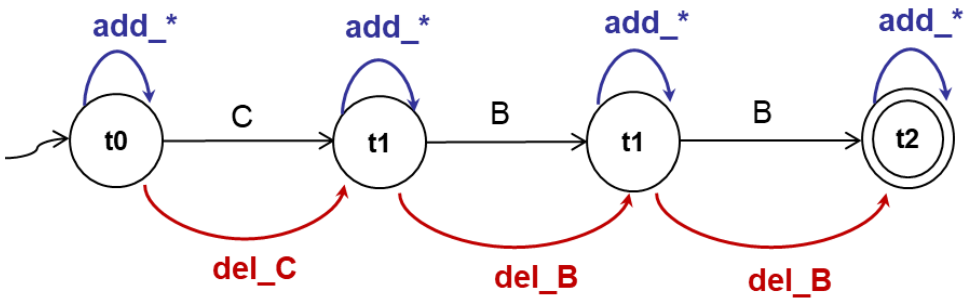


Process Model

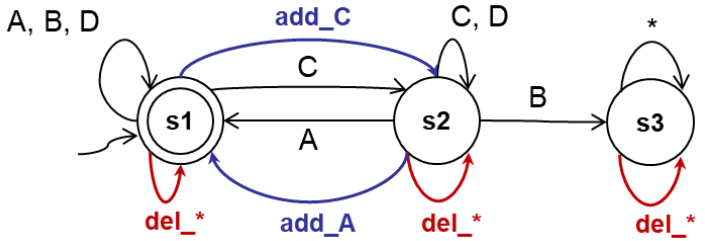


EXAMPLE OF CONTEXT-AWARE TRACE ALIGNMENT

Trace: <C,B,B>



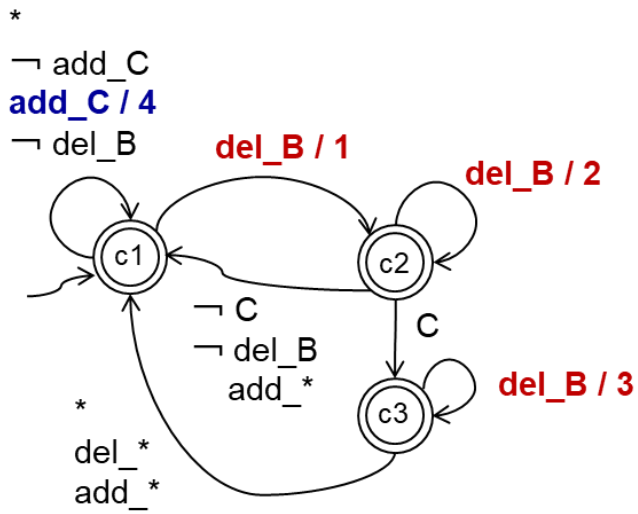
Process Model



Cost Model

Optimal Alignment (cost **4**):

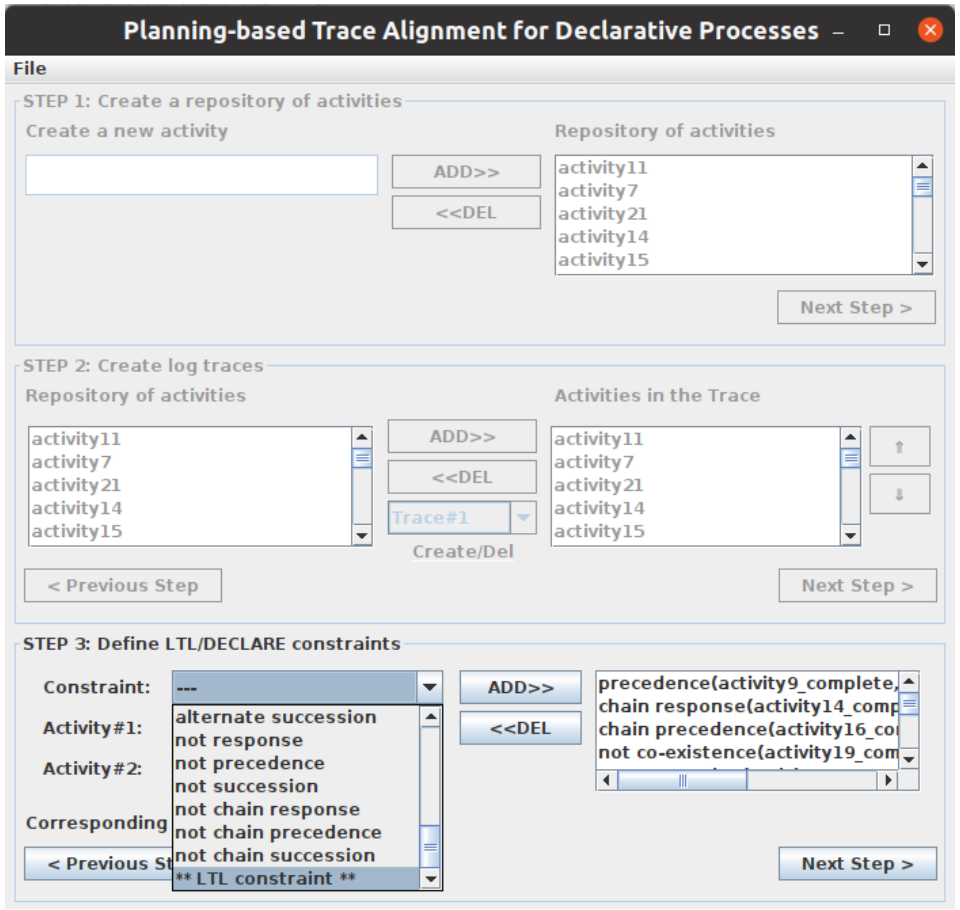
<C,delB,delB,addA>



FORMULATING TRACE ALIGNMENT IN PDDL

- Problems in PDDL are expressed in two separate parts:
  - **Planning Domain:**
    - **Planning actions** capture alignment steps in the form of:
      - synchronous moves (no cost)
      - deviations (**add** and **del** activities) to the input trace, with non-zero costs.
      - Cost depends by the specific combination of transitions in the DFAs and the cost model.
    - **Domain propositions** encode the structure and the dynamics of the augmented DFAs and the cost model.
  - **Planning Problem:**
    - **Initial state**: all DFAs in their starting state.
    - **Goal state**: all DFAs in (at least one) final state.
  - **Solution: Optimal** (**minimal-cost**) **plan** to reach the goal state.

A PLANNING-BASED TOOL FOR TRACE ALIGNMENT



3 cost-optimal state-of-the-art planners embedded:

- Fast-Downward
- Symba*-2
- Complementary1

Some **patterns** available for recurrent cost models, to mitigate the burden related to their definition.

It **accepts in input**:

- XES event logs
- Declare models, DFAs in *.dot,* LTL_f formulas

https://github.com/bpm-diag/PL_DEC_ALIGNER

EXPERIMENT SETTING AND RESULTS

Real-life Event Logs

We tested our approach with a ***real-life log*** and ***synthetic logs***.

Real-life log: incident management process

extracted from a dataset available in the UCI Machine Learning Repository, cf. http://archive.ics.uci.edu/ml/index.php.

141,712 events organized in 24,918 traces with various lengths.

Trace Length	Number of Traces	Average Preprocessing Time	Average Searching Time	Average Number of Deviations
1 – 10	388	0.90 s	0.05 s	6
11 – 20	1231	1.22 s	0.059 s	11
21 – 30	176	1.73 s	0.071 s	20
31 – 40	25	1.98 s	0.075 s	31
41 – 50	6	2.36 s	0.077 s	40
51 – 60	2	2.83 s	0.078 s	56

Performance slightly **decreases** when the trace size increases.

EXPERIMENT SETTING AND RESULTS

Synthetic Event Logs

Target: Study the **scalability** wrt **model size** and **noise in the traces**

1. We exploited the equivalence between regular languages and DFAs.
   - An LTL_f formula ϕ can be associated with a DFA that accepts exactly all traces satisfying ϕ [De-Giacomo-Vardi-IJCAI13]
2. We created 2 DFAs with the same alphabet of activities and obtained by the conjunction of 15 and 20 LTL_f formulas.
3. We specified a cost model assigning a variable cost to the repetition of some specific activities.
4. To add noise in the prefix traces, we replaced 3, 4, and 6 LTL_f formulas with their negative counterparts by obtaining 6 "noisy" DFAs.
5. For each noisy DFA, we generated 4 logs of 100 traces with different lengths, e.g., from 1 to 50 events, from 51 to 100 events, etc.

➢ **24 synthetic logs of varying complexity.**

EXPERIMENT RESULTS

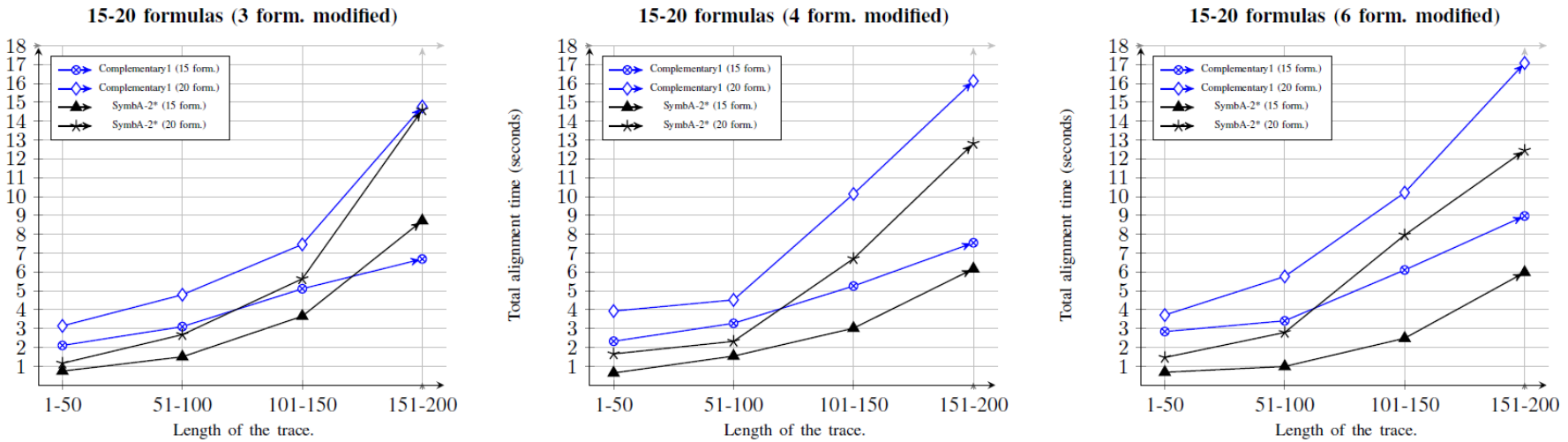
SYNTHETIC LOGS (DFA WITH 29182 STATES & 729526 TRANSITIONS)

Planners do not suffer the presence of noisy logs.

Planners scale well when length of the traces increases.

Trace length	SymbA-2* Preprocessing	SymbA-2* Searching	SymbA-2* Steps	Complementary1 Preprocessing	Complementary1 Searching	Complementary1 Steps	Context-Aware Alignment Cost	Alignment Cost
3 form. modified								
1-50	0.24	0.92	51	3.14	$2 \cdot 10^{-3}$	50	1.8	1.8
51-100	0.24	2.44	84	4.79	$4 \cdot 10^{-3}$	84	3.1	2.6
101-150	0.28	5.36	127	7.46	$6 \cdot 10^{-3}$	127	3.7	3.3
151-200	0.37	14.23	183	14.76	$9 \cdot 10^{-3}$	182	4.2	4.2
4 form. modified								
1-50	0.3	1.36	48	3.93	$2 \cdot 10^{-3}$	48	4.8	3.8
51-100	0.24	2.09	77	4.52	$3 \cdot 10^{-3}$	77	6.9	6.9
151-200	0.3	6.41	132	10.13	$7 \cdot 10^{-3}$	132	10.8	10.8
151-200	0.36	12.45	177	16.12	$9 \cdot 10^{-3}$	177	15.9	15.1
6 form. modified								
151-200	0.22	1.25	50	3.72	$2 \cdot 10^{-3}$	50	7.2	6.1
51-100	0.23	2.57	78	5.76	$4 \cdot 10^{-3}$	79		9
151-200	0.33	7.63	150	10.2	$8 \cdot 10^{-3}$	151		
151-200	0.35	12.09	184	17.07	$1.1 \cdot 10^{-2}$	185		

Planners enact around the same number of steps for alignments of the same cost.



Outline

- Introduction to Automated Planning in AI
- Automated Planning as-a-Service
- Context-Aware Trace Alignment via Automated Planning
- Concluding Remarks

CONCLUDING REMARKS

STRENGTHS

Generality: A planner can solve **arbitrary problem instances**.

- A planner does not know what the actions, and domain stand for. This is different from a *domain-specific* solver.

Scalability: Results obtained are **promising**.

- Planners efficiently cope with the size of the state space, which is exponential wrt. the size of the model, the amount of noise and the trace length.

WEAKNESS

- The price for generality is **computational.**
  - Planning is **intractable** in the worst case, yet large problems in process mining can be solved **quickly** [De-Giacomo-Marrella-AAAI17, De-Leoni-Marrella-ESWA18]

FUTURE WORK

- A Knowledge Engineering Methodology to semi-automatically convert SOC and BPM problems to planning problems.
- Go beyond classical planning.

REFERENCES

[Acitelli-ICPM22] Acitelli, G., et al. (2022) Context-Aware Trace Alignment with Automated Planning. ICPM'22

[Alman-INFSYST22] Alman, A., Maggi, F. M., Montali, M., Peñaloza, R. (2022). Probabilistic declarative process mining. Inf. Syst. 109

[De-Giacomo-Vardi-IJCAI13] De Giacomo, G., and Vardi, M. Y. (2013). Linear Temporal Logic and Linear Dynamic Logic on Finite Traces. In Proc. of IJCAI'13.

[Dumas-ACMTMIS23] Dumas, M. et al. (2023) AI-augmented Business Process Management Systems: A Research Manifesto. ACM Trans. Manag. Inf. Syst. 14(1)

[Casciani-RCIS24] Casciani, A., Bernardi, M. L., Cimitile, M., Marrella, A. (2024). Conversational Systems for AI-Augmented Business Process Management. In Proc. of RCIS'24.

[Fox-Long-JAIR03] Fox, M., Long, D. (2003) PDDL2.1: An extension to PDDL for expressing temporal planning domains. J. Artif. Intell. Res.(JAIR), vol. 20

[Geffner-M&C13] Geffner, H., Bonet, B. (2013). A Concise Introduction to Models and Methods for Automated Planning. Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan & Claypool Publishers.

[Marrella-JODS19] Marrella A. (2019). Automated Planning for Business Process Management (2019). Journal on Data Semantics, vol. 8

[van-der-Aalst-et-al-CS-R&D-09] van der Aalst, W.; Pesic, M.; and Schonenberg, H. 2009. Declarative Workflows: Balancing Between Flexibility and Support. Computer Science - R&D.

SAPIENZA
UNIVERSITÀ DI ROMA

Automated Planning-as-a-Service

Theory and Practice

Thanks for the attention