An Overview of the International Planning Competition Part 1: Classical Tracks

Amanda Coles[1] Andrew Coles[1]
Álvaro Torralba[2] Florian Pommerening[3]

[1]King's College London

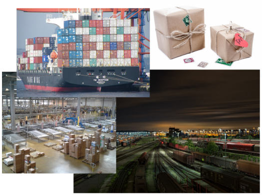
[2]Saarland University

[3]University of Basel, Switzerland

January 27, 2019

AI Planning and the IPC

Classical and Temporal Planning in a Nutshell



Classical planning

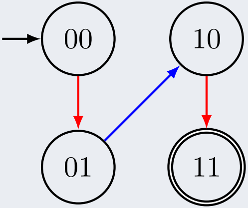
- Find operator sequence to achieve a goal
- Discrete, single-agent, observable, deterministic

Temporal planning

- actions take time and can be executed in parallel
- usually also includes numeric effects

Classical Planning Tasks

Example task: binary counter



State space

- States ○ assign values to variables
- Initial state →○
- Goal states ◎
- Operators have conditions and effects on variables

Compact Representation with PDDL

Domain

```
(define (domain trucks-example)
  (:requirements :typing)
  (:types truck location)
  (:predicates
    (CONNECTED ?from ?to - location)
    (truck-at ?t - truck ?l - location)
  )
  (:action move
    :parameters
      (?t - truck ?from ?to - location)
    :precondition
      (and (CONNECTED ?from ?to)
           (truck-at ?t ?from))
    :effect
      (and (not (truck-at ?t ?from))
           (truck-at ?t ?to))
  )
)
```

Task

```
(define (problem task1)
  (:domain trucks-example)
  (:objects
    t1 t2 - truck
    l1 l2 l3 - location
  )
  (:init
    (CONNECTED l1 l2)
    (CONNECTED l2 l3)
    (truck-at t1 l1)
    (truck-at t2 l3)
  )
  (:goal
    (and (truck-at t1 l3)
         (truck-at t2 l1))
  )
)
```

Compact Representation with PDDL

Domain

```
(define (domain trucks-example)
  (:requirements :typing)
  (:types truck location)
  (:predicates
    (CONNECTED ?from ?to - location)
    (truck-at ?t - truck ?l - location)
  )
  (:action move
    :parameters
      (?t - truck ?from ?to - location)
    :precondition
      (and (CONNECTED ?from ?to)
           (truck-at ?t ?from))
    :effect
      (and (not (truck-at ?t ?from))
           (truck-at ?t ?to))
  )
)
```

Task

```
(define (problem task1)
  (:domain trucks-example)
  (:objects
    t1 t2 - truck
    l1 l2 l3 - location
  )
  (:init
    (CONNECTED l1 l2)
    (CONNECTED l2 l3)
    (truck-at t1 l1)
    (truck-at t2 l3)
  )
  (:goal
    (and (truck-at t1 l3)
         (truck-at t2 l1))
  )
)
```

Compact Representation with PDDL

Domain

```
(define (domain trucks-example)
  (:requirements :typing)
  (:types truck location)
  (:predicates
    (CONNECTED ?from ?to - location)
    (truck-at ?t - truck ?l - location)
  )
  (:action move
    :parameters
      (?t - truck ?from ?to - location)
    :precondition
      (and (CONNECTED ?from ?to)
           (truck-at ?t ?from))
    :effect
      (and (not (truck-at ?t ?from))
           (truck-at ?t ?to))
  )
)
```

Task

```
(define (problem task1)
  (:domain trucks-example)
  (:objects
    t1 t2 - truck
    l1 l2 l3 - location
  )
  (:init
    (CONNECTED l1 l2)
    (CONNECTED l2 l3)
    (truck-at t1 l1)
    (truck-at t2 l3)
  )
  (:goal
    (and (truck-at t1 l3)
         (truck-at t2 l1))
  )
)
```

Compact Representation with PDDL

Domain

```
(define (domain trucks-example)
  (:requirements :typing)
  (:types truck location)
  (:predicates
    (CONNECTED ?from ?to - location)
    (truck-at ?t - truck ?l - location)
  )
  (:action move
    :parameters
      (?t - truck ?from ?to - location)
    :precondition
      (and (CONNECTED ?from ?to)
           (truck-at ?t ?from))
    :effect
      (and (not (truck-at ?t ?from))
           (truck-at ?t ?to))
  )
)
```

Task

```
(define (problem task1)
  (:domain trucks-example)
  (:objects
    t1 t2 - truck
    l1 l2 l3 - location
  )
  (:init
    (CONNECTED l1 l2)
    (CONNECTED l2 l3)
    (truck-at t1 l1)
    (truck-at t2 l3)
  )
  (:goal
    (and (truck-at t1 l3)
         (truck-at t2 l1))
  )
)
```

Compact Representation with PDDL

Domain

```
(define (domain trucks-example)
  (:requirements :typing)
  (:types truck location)
  (:predicates
    (CONNECTED ?from ?to - location)
    (truck-at ?t - truck ?l - location)
  )
  (:action move
    :parameters
      (?t - truck ?from ?to - location)
    :precondition
      (and (CONNECTED ?from ?to)
           (truck-at ?t ?from))
    :effect
      (and (not (truck-at ?t ?from))
           (truck-at ?t ?to))
  )
)
```

Task

```
(define (problem task1)
  (:domain trucks-example)
  (:objects
    t1 t2 - truck
    l1 l2 l3 - location
  )
  (:init
    (CONNECTED l1 l2)
    (CONNECTED l2 l3)
    (truck-at t1 l1)
    (truck-at t2 l3)
  )
  (:goal
    (and (truck-at t1 l3)
         (truck-at t2 l1))
  )
)
```

Compact Representation with PDDL

Domain

```
(define (domain trucks-example)
  (:requirements :typing)
  (:types truck location)
  (:predicates
    (CONNECTED ?from ?to - location)
    (truck-at ?t - truck ?l - location)
  )
  (:action move
    :parameters
      (?t - truck ?from ?to - location)
    :precondition
      (and (CONNECTED ?from ?to)
           (truck-at ?t ?from))
    :effect
      (and (not (truck-at ?t ?from))
           (truck-at ?t ?to))
  )
)
```

Task

```
(define (problem task1)
  (:domain trucks-example)
  (:objects
    t1 t2 - truck
    l1 l2 l3 - location
  )
  (:init
    (CONNECTED l1 l2)
    (CONNECTED l2 l3)
    (truck-at t1 l1)
    (truck-at t2 l3)
  )
  (:goal
    (and (truck-at t1 l3)
         (truck-at t2 l1))
  )
)
```

Compact Representation with PDDL

Domain

```
(define (domain trucks-example)
  (:requirements :typing)
  (:types truck location)
  (:predicates
    (CONNECTED ?from ?to - location)
    (truck-at ?t - truck ?l - location)
  )
  (:action move
    :parameters
      (?t - truck ?from ?to - location)
    :precondition
      (and (CONNECTED ?from ?to)
           (truck-at ?t ?from))
    :effect
      (and (not (truck-at ?t ?from))
           (truck-at ?t ?to))
  )
)
```

Task

```
(define (problem task1)
  (:domain trucks-example)
  (:objects
    t1 t2 - truck
    l1 l2 l3 - location
  )
  (:init
    (CONNECTED l1 l2)
    (CONNECTED l2 l3)
    (truck-at t1 l1)
    (truck-at t2 l3)
  )
  (:goal
    (and (truck-at t1 l3)
         (truck-at t2 l1))
  )
)
```

Compact Representation with PDDL

Domain

```
(define (domain trucks-example)
  (:requirements :typing)
  (:types truck location)
  (:predicates
    (CONNECTED ?from ?to - location)
    (truck-at ?t - truck ?l - location)
  )
  (:action move
    :parameters
      (?t - truck ?from ?to - location)
    :precondition
      (and (CONNECTED ?from ?to)
           (truck-at ?t ?from))
    :effect
      (and (not (truck-at ?t ?from))
           (truck-at ?t ?to))
  )
)
```

Task

```
(define (problem task1)
  (:domain trucks-example)
  (:objects
    t1 t2 - truck
    l1 l2 l3 - location
  )
  (:init
    (CONNECTED l1 l2)
    (CONNECTED l2 l3)
    (truck-at t1 l1)
    (truck-at t2 l3)
  )
  (:goal
    (and (truck-at t1 l3)
         (truck-at t2 l1))
  )
)
```

Compact Representation with PDDL

Domain

```
(define (domain trucks-example)
  (:requirements :typing)
  (:types truck location)
  (:predicates
    (CONNECTED ?from ?to - location)
    (truck-at ?t - truck ?l - location)
  )
  (:action move
    :parameters
      (?t - truck ?from ?to - location)
    :precondition
      (and (CONNECTED ?from ?to)
           (truck-at ?t ?from))
    :effect
      (and (not (truck-at ?t ?from))
           (truck-at ?t ?to))
  )
)
```

Task

```
(define (problem task1)
  (:domain trucks-example)
  (:objects
    t1 t2 - truck
    l1 l2 l3 - location
  )
  (:init
    (CONNECTED l1 l2)
    (CONNECTED l2 l3)
    (truck-at t1 l1)
    (truck-at t2 l3)
  )
  (:goal
    (and (truck-at t1 l3)
         (truck-at t2 l1))
  )
)
```

Compact Representation with PDDL

Domain

```
(define (domain trucks-example)
  (:requirements :typing)
  (:types truck location)
  (:predicates
    (CONNECTED ?from ?to - location)
    (truck-at ?t - truck ?l - location)
  )
  (:action move
    :parameters
      (?t - truck ?from ?to - location)
    :precondition
      (and (CONNECTED ?from ?to)
           (truck-at ?t ?from))
    :effect
      (and (not (truck-at ?t ?from))
           (truck-at ?t ?to))
  )
)
```

Task

```
(define (problem task1)
  (:domain trucks-example)
  (:objects
    t1 t2 - truck
    l1 l2 l3 - location
  )
  (:init
    (CONNECTED l1 l2)
    (CONNECTED l2 l3)
    (truck-at t1 l1)
    (truck-at t2 l3)
  )
  (:goal
    (and (truck-at t1 l3)
         (truck-at t2 l1))
  )
)
```

Evolution of PDDL

- 1998, 2000: PDDL 1.0: STRIPS, ADL (quantified effects and preconditions, conditional effects)
- 2002: PDDL 2.1: temporal + numeric planning
- 2004: PDDL 2.2: derived predicates and timed initial literals
- 2006: PDDL 3.0: soft goals and state trajectory constraints
- 2008: Restricted PDDL Features: STRIPS + action costs
- 2014: Re-introduced conditional effects

Goals of the IPC

Goals

- evaluate state-of-the-art planning systems
- promote planning research
- highlight challenges
- provide new benchmarks

IPC Organization

Organization

- different tracks for different planning variants
- tracks organized more or less independently
  - initiative of track organizers
  - IPC happens if someone organizes a track

Organizing a Track

Jobs as an organizer

- track rules
- benchmarks
  - create/elicit new domains
  - select instances
  - find reference solutions
- participants
  - elicit participation
  - compile planners
  - assist in testing/bug fixing
- experiments
  - run planners on benchmarks
  - evaluate results

IPC Tracks

Classical Planning Tracks

- satisficing (1998, 2000, 2004, 2006, 2008, 2011, 2014, 2018)
- optimal (2004, 2006, 2008, 2011, 2014, 2018)
- satisficing multi-core (2011, 2014)
- agile (2014, 2018)
- cost-bounded (2018)

Temporal Metric Planning

- satisficing (2002, 2004, 2008, 2011, 2014, 2018)
- optimal (2006, 2008, 2014)
- agile (2018)

...

IPC Tracks (continued)

Probabilistic Planning

- MDP (2004, 2006, 2011, 2018)
- conformant (2006, 2008)
- POMDP (2011)
- FOND, NOND (2008)
- continuous (2014)

Preferences, Constraints, Net-benefit

- satisficing (2006, 2008, 2014)
- optimal (2008, 2014)

Learning (2008, 2011, 2014)
Unsolvability (2016)
Hand-Tailored, Domain-Specific tracks (1998, 2000, 2002)

Classical Tracks

Classical Tracks

Classical Planning:

- Deterministic and Fully-observable environment
- Find a sequence of actions that leads to the goal

Several Tracks:

- Optimal Track: find a plan of minimum cost
- Satisficing Track: find a plan as good as possible (but not necessarily optimal)
- Agile Track: find a plan as quickly as possible
- Cost-Bounded Track: find a plan whose cost is below a bound

Benchmarks

How to evaluate a general solver?

- The goal in planning is to develop a decision-making tool that can work in any situation

How to evaluate a general solver?

- The goal in planning is to develop a decision-making tool that can work in any situation
- But we evaluate it in concrete situations!

How to evaluate a general solver?

- The goal in planning is to develop a decision-making tool that can work in any situation
- But we evaluate it in concrete situations!
- Different planners may do best on different situations so a "good" benchmark selection is essential for the competition.
- Ideally benchmarks should:
  - be diverse: so that planners are evaluated in different scenarios avoiding "overfitting" to a particular class of planning problems
  - be inspired in real-world problems: so that the evaluation targets cases that are relevant for real-world applications
  - be challenging: so that research can be conducted on how to extend the planners to be effective in more scenarios

IPC Benchmarks

Benchmarks published in each IPC:

- IPC 1998: assembly, gripper, logistics, movie, mprime, mystery
- IPC 2000: blocks, elevators, freecell, logistics, schedule
- IPC 2002: depot, driverlog, freecell, rovers, satellite, zenotravel
- IPC 2004: airport, optical-telegraphs, philosophers, pipesworld, psr-large, psr-middle, psr-small
- IPC 2006: openstacks, pathways, pipesworld, rovers, storage, tpp, trucks
- IPC 2008: cybersec, elevators, openstacks, parcprinter, pegsol, scanalyzer, sokoban, transport, woodworking
- IPC 2011: barman, elevators, nomystery, openstacks, parcprinter, parking, pegsol, scanalyzer, sokoban, tidybot, transport, visitall, woodworking
- IPC 2014: barman, cavediving, childsnack, citycar, floortile, ged, hiking, maintenance, openstacks, parking, tetris, thoughtful, tidybot, transport, visitall

All of them publicly available to evaluate new planning algorithms!
`http://planning.domains`

New domains in 2018

PDDL features

- no new PDDL feature this time but ...
- ...stronger focus on conditional effects and grounding

⇝ In 2 domains, we used two different formulations using the tool by Bustos et al., (2014)

New domains in 2018

PDDL features

- no new PDDL feature this time but . . .
- . . . stronger focus on conditional effects and grounding

⇝ In 2 domains, we used two different formulations using the tool by Bustos et al., (2014)

Competition Domains

- 11 new domains
  - 5 from planning applications
  - no domains from previous IPCs
- not all domains used in all tracks
  - Optimal/Satisficing/Agile: 10 domains
  - Cost-bounded: 8 domains

New domains in 2018

PDDL features

- no new PDDL feature this time but . . .
- . . . stronger focus on conditional effects and grounding

⇝ In 2 domains, we used two different formulations using the tool by Bustos et al., (2014)

Competition Domains

- 11 new domains
  - 5 from planning applications
  - no domains from previous IPCs
- not all domains used in all tracks
  - Optimal/Satisficing/Agile: 10 domains
  - Cost-bounded: 8 domains

Domain Submissions

- "Thank you!" to everyone who submitted a domain
- more submissions than we could handle

Agricola

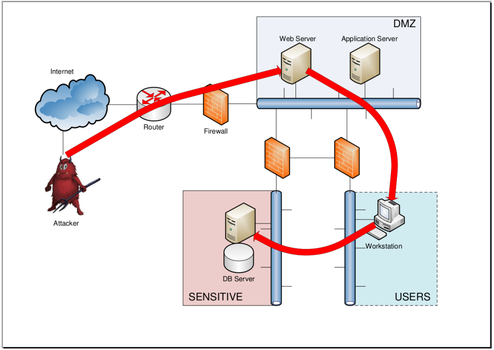
Submitted by: Tomás de la Rosa, Universidad Carlos III de Madrid

Loosely based on the board game "Agricola".

$\leadsto$ dead-ends

Caldera

Submitted by: Andy Applebaum, Doug Miller, and Blake Strom, MITRE.

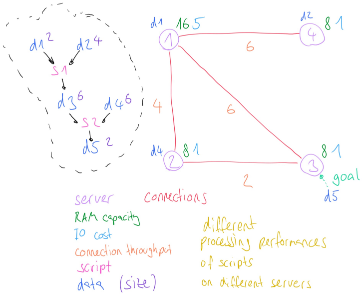


Cybersecurity domain based on a real-world application.

- Delete-free domain
- Quantified Conditional Effects
- Hard to ground

Data Network

Submitted by: Submitted by: Manuel Heusner, Basel University



Process and send data accross a computer network.

⤳ Our Logistics variant ☺

Flash Fill

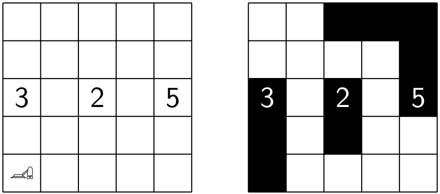
Submitted by: Javier Segovia, Universitat Pompeu Fabra

	A	B
1	Florian Pommerening	F P
2	Alvaro Torralba	A T
3	Foo Bar	F B
4	Bar Foo	
5		
6		
7		

Excel Flashfill feature modelled as a classical planning problem by using the planning programs compilation by Segovia et al.

⇝ Quantified conditional effects (hard to handle)

Nurikabe



Version of Floortile where the robot must decide the painting pattern

$\leadsto$ Quantified conditional effects (easy to handle)

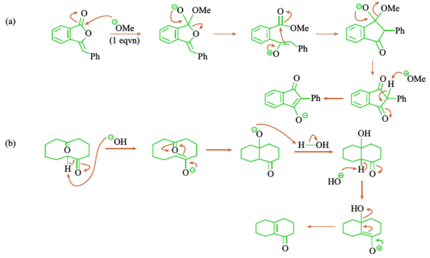
Organic Synthesis

Submitted by: Hadi Qovaizi, Arman Masoumi, Anne Johnson, Russell Viirre, Andrew McWilliams, and Mikhail Soutchanski, Ryerson University

Massachusetts Institute of Technology

5.13: Organic Chemistry II

11.

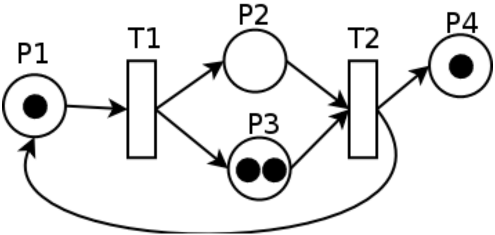


Find a sequence of reactions that produces the target molecule from given initial molecules. The instances are based on real exam questions.

⤳ Hard to ground

Petri Net Alignment

Submitted by: Massimiliano de Leoni and Andrea Marrella, Eindhoven University of Technology



Align the execution of a petri net to a sequence of events

⇝ 0-cost actions

Settlers

Submitted by: Marcel Steinmetz, Saarland University

Resource-constrained version of the numeric domain Settlers

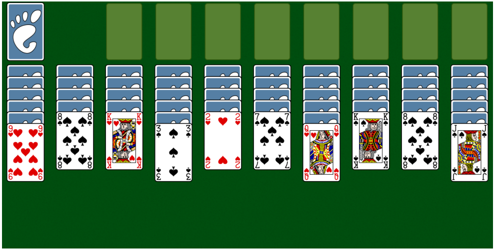
⇝ Quantified conditional effects (hard to compile away)

Snake

Version of the Snake game where the location where apples will spawn is known in advance.

⤳ Many facts (snake representation)

Spider



Variant of the Spider card game where all cards are faced up from the beginning.

- ⤳ Conditional effects
- ⤳ 0-cost actions

Termes

Submitted by: Sven Koenig and Satish Kumar

Single agent variant of Harvard TERMES robots, based on termites.

⤳ Long plans

Overview

	Agricola	Caldera	Data Network	Flashfill	Nurikabe	Organic Synthesis	Petri Net Alignment	Settlers	Snake	Spider	Termes	SUM
C. Eff		✓!		✓!!	✓			✓!!		✓		5
0 cost							✓			✓		2
costs	✓	?	✓	✓		?		✓				4-6

- 4 domains from “applications” (not developed for the IPC): Caldera, Flashfill, Organic Synthesis, Petri-net Alignment

Solving Classical Planning Tasks

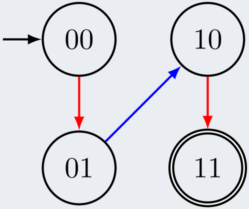
Solving Classical Planning Tasks: Search

Two important approaches

- explicit state search (A*, GBFS, ...)
  - every search node represents a state
  - expansion: generating successors for applicable operators
  - search guided by heuristic
- symbolic seach
  - every search node represents a set of states
  - expansion: generating all states reachable in one step
  - sets of states compactly represented (BDD, ...)
  - can also be guided by heuristic

Solving Classical Planning Tasks: Abstractions

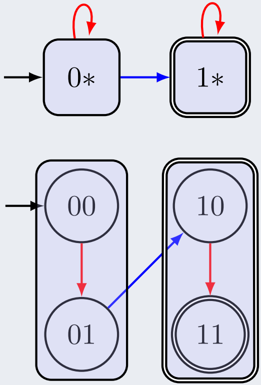
Abstractions of Planning Tasks



- full state space too big
  - example: plan for 10 trucks in 10 cities
- map to smaller space
- extract lower bound from abstractions

Solving Classical Planning Tasks: Abstractions

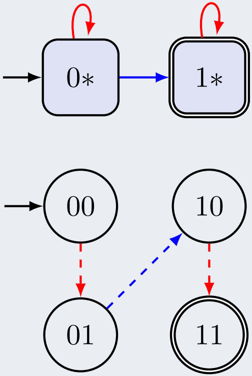
Abstractions of Planning Tasks



- full state space too big
  - example: plan for 10 trucks in 10 cities
- map to smaller space
- extract lower bound from abstractions

Solving Classical Planning Tasks: Abstractions

Abstractions of Planning Tasks



- full state space too big
  - example: plan for 10 trucks in 10 cities
- map to smaller space
- extract lower bound from abstractions

Solving Classical Planning Tasks: Delete Relaxations

Domain

```
(:action move
  :parameters
    (?t - truck ?from ?to -
location)
  :precondition
    (and (CONNECTED ?from ?to)
         (truck-at ?t ?from))
  :effect
    (and (not (truck-at ?t ?from))
         (truck-at ?t ?to))
)
```

Delete Relaxations

- modify domain so deleting a fact never helps
- ignore some or all delete effects
- problem is simpler to solve
- heuristic value: solution cost in the relaxation

Solving Classical Planning Tasks: Delete Relaxations

Domain

```
(:action move
  :parameters
    (?t - truck ?from ?to -
location)
  :precondition
    (and (CONNECTED ?from ?to)
         (truck-at ?t ?from))
  :effect
    (and (not (truck-at ?t ?from))
         (truck-at ?t ?to))
)
```

Delete Relaxations

- modify domain so deleting a fact never helps
- ignore some or all delete effects
- problem is simpler to solve
- heuristic value: solution cost in the relaxation

Solving Classical Planning Tasks: Delete Relaxations

Domain

```
(:action move
  :parameters
    (?t - truck ?from ?to -
location)
  :precondition
    (and (CONNECTED ?from ?to)
         (truck-at ?t ?from))
  :effect
    (and ~~(not (truck-at ?t ?from))~~
         (truck-at ?t ?to))
)
```

Delete Relaxations

- modify domain so deleting a fact never helps
- ignore some or all delete effects
- problem is simpler to solve
- heuristic value: solution cost in the relaxation

Solving Classical Planning Tasks: Novelty

Novelty

- when exploring the state space prefer new areas
- a state is novel if we see parts of it for the first time
- the more general the part, the more novel the state
- limit search to only explore novel states
- can be combined with heuristics (best-first width search)

IPC

The International Planning Competition (IPC)

- semi-regular competition
  - 1998, 2000, 2002, 2004, 2006, 2008, 2011, 2014, 2018
- organized in the context of the International Conference on Planning and Scheduling (ICAPS)

Past and Future IPCs

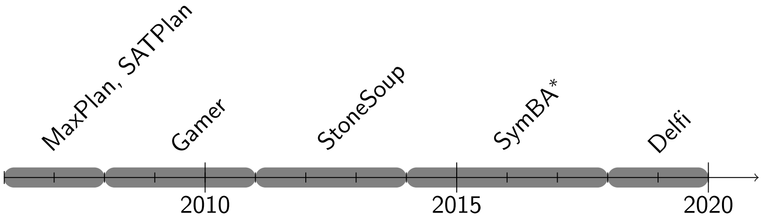
- `icaps-conference.org/index.php/Main/Competitions`
- `icaps-conference@googlegroups.com`

Optimal Track

Rules of the Optimal Track

- Goal: Find an optimal plan
- Metric: number of plans solved

Trends and Breakthroughs: Optimal Planning



- SAT planners (MaxPlan, SATPlan)
- Symbolic Search planners (Gamer, SymBA*)
- Heuristic search planners
- Portfolios (StoneSoup, Delfi)

Techniques used in 2018

- abstraction heuristics
    - many and most sucessful submissions
- landmark heuristics
- critical path heuristics
- decoupled search
- symbolic search
    - hard-to-beat baseline: blind symbolic bi-directional search

Coverage	agricola	caldera	data-net.	flashfill	nurikabe	org.-syn.	settlers	snake	spider	termes	SUM
Delfi1	12	13	13	12	13	**20**	9	11	11	12	126
Complementary2	6	12	12	12	13	18	9	**14**	12	16	124
Complementary1	10	11	**14**	**13**	13	17	8	11	11	16	124
Planning-PDBs	6	12	**14**	11	13	18	8	13	11	16	122
symb. Bi-dir.	**15**	10	13	11	13	19	8	4	7	**18**	118
Scorpion	2	12	**14**	**13**	13	0	**10**	**14**	**17**	14	109
Delfi2	11	11	13	11	13	9	8	7	7	15	105
FDMS2	14	12	9	12	13	2	8	11	11	12	104
FDMS1	9	12	10	12	13	2	9	11	11	12	101
DecStar	0	8	**14**	11	**14**	8	8	11	13	12	99
Metis1	0	13	12	12	**14**	9	9	7	11	6	93
MSP	7	8	13	8	12	10	0	11	6	16	91
Metis2	0	**15**	12	12	**14**	9	0	7	12	6	87
Blind	0	8	7	11	10	7	8	12	11	10	84
Symple-2	1	8	9	7	13	2	0	0	5	13	58
Symple-1	0	8	9	8	13	2	0	0	4	13	57
maplan-2	2	2	9	0	6	0	0	**14**	1	12	46
maplan-1	0	2	12	0	6	0	0	7	10	6	43

	A^*	Symbolic	PDB	M&S	CEGAR	LM-cut	h^m	Symmetry	POR	# best	Score
Delfi1	✓	✓	✓	✓		✓		✓	✓	1	126
Complementary2	✓		✓							1	124
Complementary1	✓		✓							2	124
Planning-PDBs	✓		✓							1	122
symb. Bi-dir.		✓								2	118
Scorpion	✓				✓					5	109
Delfi2	✓	✓	✓							0	105
FDMS2	✓			✓						0	104
FDMS1	✓			✓						0	101
DecStar	✓					✓		✓	✓	2	99
Metis1	✓					✓		✓	✓	1	93
MSP	✓	✓				✓				0	91
Metis2	✓					✓		✓	✓	2	87
Blind	✓									0	84
Symple-2		✓								0	58
Symple-1		✓								0	57
maplan-2	✓						✓			1	46
maplan-1	✓						✓			0	43

Conclusions: Optimal Track

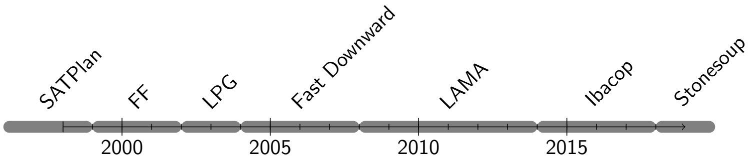
- Lots of research done in abstraction heuristics has paid off: PDBs, CEGAR, M&S
- A portfolio won the track but non-portfolio planners are still very competitive
- Symbolic search and A^* are two competitive approaches for optimal planning

Satisficing Track

Rules of the Satisficing Track

- Goal: Find a plan with high quality
- Metric: $C/C*$
  - same as in 2008 but different from 2011, 2014
  - reference plans by many different means

Trends and Breakthroughs: Satisficing Planning



- SAT-based planners (SAT-plan, Madagascar)
- Heuristic search planners (FF, LPG, Fast Downward, LAMA)
- Portfolios (Ibacop, Stonesoup)

Techniques used in 2018

- delete-relaxation heuristics
  - many variants of partial delete relaxation
- decoupled search

Sat score	agricola	caldera	data-net.	flashfill	nurikabe	org.-syn.	settlers	snake	spider	termes	SUM
Stone Soup	13	14	10	13	18	9	**16**	7	10	8	123
Remix	13	14	10	12	**18**	9	16	7	10	6	120
DUAL-BFWS	12	**17**	11	**16**	14	11	6	9	12	5	119
Saarplan	**14**	11	12	13	16	11	9	8	10	7	116
DecStar	12	13	10	13	12	9	15	4	12	6	111
Cerberus	10	10	11	9	15	**12**	9	5	13	7	108
LAMA 2011	9	13	7	13	10	**12**	15	3	13	7	107
BFWS-Pref.	11	15	8	11	12	7	8	**15**	10	5	106
Cerberus-gl	10	10	11	9	15	**12**	9	5	**14**	6	106
OLCFF	13	11	12	0	17	9	0	7	11	7	92
POLY-BFWS	13	17	11	8	10	5	9	2	8	2	90
IBaCoP	10	5	**14**	0	6	8	0	8	8	**10**	73
IBaCoP2	11	6	11	0	7	8	0	7	7	7	66
MERWIN	10	0	10	0	5	**12**	0	4	11	7	62
mercury	12	0	8	0	5	11	0	3	12	7	61
DFS+	10	12	6	1	5	5	7	4	7	0	60
fs-sim	11	6	5	0	10	4	0	7	4	3	53
fs-blind	3	6	5	0	12	4	0	7	5	7	50
freelunch-dr	8	1	0	0	0	6	0	5	0	0	22
freelunch-ma	0	2	2	0	3	8	0	1	0	0	16
Symple-2	1	2	0	0	2	5	0	0	0	1	11
Symple-1	1	2	0	0	2	5	0	0	0	1	11
alien	4	0	1	0	0	4	0	0	0	0	9

Sat score	GBFS	EHC	SAT	hFF	hRB	hCFF	Lm	Nov	# best	Score
FF	✓	✓		✓						
LPG				✓						
Fast Downward	✓			✓						
LAMA 2011	✓			✓			✓		1	107
IbaCop 2014	✓		✓	✓			✓			
Stone Soup	✓			✓			✓		1	123
Remix	✓			✓			✓		1	120
DUAL-BFWS	✓			✓			✓	✓	2	119
Saarplan	✓	✓		✓	✓	✓	✓		1	116
DecStar	✓			✓			✓		0	111
Cerberus | -gl	✓				✓		✓	✓	1 | 2	108 | 106
BFWS-Pref.	✓			✓			✓	✓	1	106
OLCFF	✓	✓		✓		✓	✓	✓	0	92
POLY-BFWS	✓							✓	0	90
IBaCoP 1–2	✓		✓	✓	✓		✓		2	73 | 66
MERWIN	✓				✓		✓	✓	1	62
mercury	✓				✓		✓		0	61
DFS+	✓							✓	0	60
fs-sim	✓							✓	0	53
fs-blind	✓							✓	0	50
freelunch-dr | -ma			✓						0	22
Symple-1 | 2									0	11 | 11
alien	✓			✓					0	9

Conclusions: Satisficing Track

- hFF still very relevant today: top 8 planners use it or a variant thereof (red-black or CFF)
- Many planners using different variants of novelty
- best-first width search
  - best single-planner performance
  - very agile
- SAT planning
  - entries not competitive on the 2018 domains

Agile Track

Rules of the Agile Track

- Goal: Find a plan quickly
- Metric: $1 - \log(t)/\log(300)$, or 1 if solved in first second
  - different from 2014
  - independent of reference time
  - stronger emphasis on solving in short time
- Instance selection:
  - Same instances as in satisficing track

Agile Track

- Recently introduced in 2014
- Techniques similar to those for satisficing planning

Agl score	**agricola**	**caldera**	**data-net.**	**flashfill**	**nurikabe**	**org.-syn.**	**settlers**	**snake**	**spider**	**termes**	**SUM**
BFWS-Pref.	2.3	6.9	6.1	**8.8**	**7.5**	3.9	2.4	8.9	**5.6**	3.8	56.1
LAMA 2011	0.8	6.6	7.6	7.4	6.3	2.9	**7.1**	2.0	4.6	**7.6**	52.7
Saarplan	1.4	6.6	**9.5**	3.4	7.5	1.9	3.8	3.8	2.0	6.3	46.3
DUAL-BFWS	1.6	**7.6**	4.4	8.0	7.1	**4.8**	1.7	3.8	4.2	3.1	46.2
Remix	1.2	6.1	6.6	6.0	7.1	3.3	5.6	1.6	1.5	5.4	44.3
POLY-BFWS	2.2	7.5	5.4	6.7	7.4	2.8	1.8	2.5	4.8	1.9	43.0
DecStar	1.4	5.8	5.3	3.9	6.4	2.3	5.6	1.8	2.6	6.3	41.4
OLCFF	1.3	6.6	9.1	0.4	7.4	1.7	0.0	3.8	1.7	6.0	38.1
Cerberus	0.5	5.9	4.8	2.4	7.4	1.5	1.7	2.7	0.7	6.8	34.4
Cerberus-gl	0.4	5.8	4.8	2.4	7.5	1.6	1.7	2.6	0.7	3.6	31.0
LAPKT-DFS+	2.4	6.6	1.9	0.3	4.1	2.0	2.6	0.7	3.5	0.0	24.1
mercury2014	1.1	0.0	7.1	0.0	1.1	2.5	0.0	1.9	3.4	6.4	23.5
fs-blind	0.5	3.4	2.4	0.0	7.4	0.2	0.0	4.7	1.5	3.4	23.5
fs-sim	**2.5**	3.3	3.2	0.0	6.5	0.4	0.0	2.7	1.1	3.0	22.8
MERWIN	0.9	0.0	7.0	0.0	1.1	2.5	0.0	1.8	2.8	5.7	21.7
freelunch-dr	1.1	0.9	3.4	0.0	0.0	1.2	0.0	**10.8**	1.8	0.0	19.2
IBaCoP	0.3	0.4	1.5	0.0	0.1	1.0	0.0	0.4	0.0	0.8	4.5
freelunch-ma	0.0	1.4	0.8	0.0	0.6	1.0	0.0	0.0	0.0	0.0	3.9
alien	0.6	0.0	1.1	0.0	0.0	1.8	0.0	0.0	0.0	0.0	3.5
IBaCoP2	0.3	0.1	1.1	0.0	0.0	0.5	0.0	0.4	0.0	0.6	3.0
Symple-2	0.0	0.1	0.0	0.0	0.4	1.7	0.0	0.0	0.0	0.0	2.1
Symple-1	0.0	0.1	0.0	0.0	0.5	1.4	0.0	0.0	0.0	0.0	2.1

Agl score	GBFS	EHC	SAT	hFF	hRB	hCFF	Lm	Nov	# best	Score
BFWS-Pref.	✓			✓			✓	✓	3	56.1
LAMA 2011	✓			✓			✓		2	52.7
Saarplan	✓	✓		✓	✓	✓	✓		1	46.3
DUAL-BFWS	✓			✓			✓	✓	2	46.2
Remix	✓			✓			✓			44.3
POLY-BFWS	✓							✓		43.0
DecStar	✓			✓			✓			41.4
OLCFF	✓	✓		✓		✓	✓	✓		38.1
Cerberus | -gl	✓				✓		✓	✓		34.4 | 31.0
LAPKT-DFS+	✓							✓		24.1
mercury2014	✓				✓		✓			23.5
fs-blind	✓							✓		23.5
fs-sim	✓							✓	1	22.8
MERWIN	✓				✓		✓	✓		21.7
freelunch-dr | -ma			✓						1	19.2 | 3.9
IBaCoP 1–2	✓		✓	✓	✓		✓			4.5 | 3.0
alien	✓			✓						3.5
Symple-1 | 2										2.1

Conclusions: Agile Track

- Portfolios not dominant when a solution needs to be found quickly
- LAMA is still a very strong competitor
  - very stable on domains with conditional effects
- Best-first width search was a very dominant approach

Cost-Bounded Track

Cost-bounded Track

- Goal: Find a plan with costs below given bound
- Metric: number of plans solved
- Instance selection:
  - mix of instances from satisficing and optimal track
- Bound selection:
  - Very Tight: find an optimal solution (similar to the optimal track but there is no need to prove that it is optimal)
  - Very Loose: find any solution (similar to the agile track)
- To keep things interesting we used two tight bounds per instance

Techniques in the Cost-Bounded Track

- Most planners are configurations from either the optimal or the agile track adapted to return only solutions with a valid cost.

Coverage	**agricola**	**caldera**	**data-net.**	**nurikabe**	**settlers**	**snake**	**spider**	**termes**	**SUM**
Stone Soup	8	16	8	**15**	**14**	9	9	**8**	87
Remix	8	13	**10**	14	**14**	**10**	**10**	7	86
Saarplan	**9**	12	7	11	10	9	7	**8**	73
DUAL-BFWS	6	**17**	9	6	7	**10**	6	3	64
LAMA 2011	8	9	9	7	10	5	6	**8**	62
Complementary2	8	10	3	10	5	**10**	9	6	61
Cerberus-gl	3	8	7	11	9	8	7	5	58
Cerberus	2	8	7	11	9	8	7	5	57
Planning-PDBs	5	7	3	9	5	7	**10**	6	52
DecStar	5	8	2	10	8	5	8	5	51
Complementary1	6	8	2	12	4	5	8	6	51
OLCFF	5	12	4	12	0	**10**	5	1	49
MERWIN	8	0	9	4	0	6	4	3	34
Symple-2	0	2	0	2	0	0	2	4	10
Symple-1	0	2	0	2	0	0	2	4	10

Coverage	**agricola**	**caldera**	**data-net.**	**nurikabe**	**settlers**	**snake**	**spider**	**termes**	**SUM**
Stone Soup	8	16	8	**15**	**14**	9	9	**8**	87
Remix	8	13	**10**	14	**14**	**10**	**10**	7	86
Saarplan	**9**	12	7	11	10	9	7	**8**	73
DUAL-BFWS	6	**17**	9	6	7	**10**	6	3	64
LAMA 2011	8	9	9	7	10	5	6	**8**	62
Complementary2	8	10	3	10	5	**10**	9	6	61
Cerberus-gl	3	8	7	11	9	8	7	5	58
Cerberus	2	8	7	11	9	8	7	5	57
Planning-PDBs	5	7	3	9	5	7	**10**	6	52
DecStar	5	8	2	10	8	5	8	5	51
Complementary1	6	8	2	12	4	5	8	6	51
OLCFF	5	12	4	12	0	**10**	5	1	49
MERWIN	8	0	9	4	0	6	4	3	34
Symple-2	0	2	0	2	0	0	2	4	10
Symple-1	0	2	0	2	0	0	2	4	10

Conclusions: Cost-bounded Track

- Portfolios clearly dominate non-portfolio approaches
- Satisficing planning techniques are generally stronger than optimal planning techniques
  - → Even if the bound is the optimal solution cost!
- Great margin of improvement on designing specific algorithms for cost-bounded planning.

Summary

What the IPC 2018 brought us

- New domains with interesting challenges:
  - Hard to ground benchmarks
  - Domains with heavy use of conditional effects
- New planning algorithms
  - Stronger abstraction heuristics: PDBs, CEGAR, M&S, . . .
  - Novelty
  - Decoupled Search
  - Comeback of Enforced Hill Climbing

Portfolios

- winners of 3/4 tracks
- recent trend, also in other competitions
- avoid weaknesses of single planners
- well suited for exponential scaling of benchmarks

Controversy

- complaints about attribution and interpretability
- move to separate track?
  - hard to clearly define (e.g., LAMA)

  $\rightsquigarrow$ Sparkle Planning Challenge 2019

Get Involved

Write a Planner

Have an idea for a new technique?

Many tools available

- domains: `planning.domains`, `bitbucket.org/aibasel`
- translator: `fast-downward.org`
- planning framework: `fast-downward.org`
- validator: `github.com/KCL-Planning/VAL`, `github.com/patrikhaslum/INVAL`

Demo: Add a New Heuristic to Fast Downward

Submit a Planner

Want to submit your planner?

- different submission procedures over the years
- container technology used in 2018: Singularity

⇝ containerized versions of all 2018 participants available

Demo: Add a Singularity Script to Fast Downward

Organize an IPC Track

Interested in a track?

- Organize it!
- Don't wait for the next "classical" track.
- Get in touch
  - ICAPS competition liaison (Scott Sanner)
  - previous organizers like us (`ipc2018.bitbucket.io`)

Contribute to the IPC Workshop

IPC Workshop at ICAPS 2019

- result analyses
- track/rule suggestions
- opinion papers
- benchmarks
- metrics
- tools

Format

- 30/15/5 minutes presentations
- discussions

The Temporal Track of the International Planning Competition

Amanda Coles and Andrew Coles

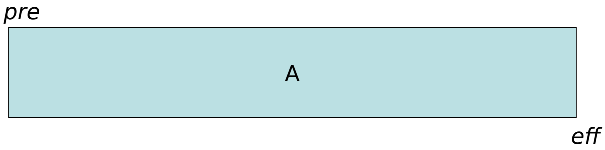
King’s College London, UK

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement No. 730086 (ERGO).

Temporal Planning

- In general, activities have **varying durations**:
  - Loading a package onto a truck is much quicker than driving the truck;
  - Drinking a cup of tea takes longer than making it;
  - Procrastinating tasks takes longer than doing them;
  - ...

TGP Durative Actions



- All Preconditions must hold at the start of the action;
- Preconditions that do not appear in effects must hold throughout execution;
- Effects are undefined during execution and only guaranteed to hold at the final time point.

"Temporal Planning with Mutual Exclusion Reasoning" D. Smith & D. Weld, IJCAI 1999.

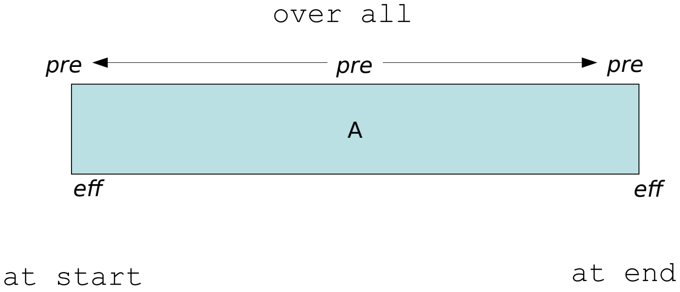
Temporal Graph Plan

- Using the action model described above;
- Modified version of Graphplan;
- Makespan optimal;
- Also capable of reasoning about exogenous events/time windows (TILs).

"Temporal Planning with Mutual Exclusion Reasoning" D. Smith & D. Weld, IJCAI 1999.

Durative Actions in PDDL 2.1

First Temporal Track @ Third IPC: 2002



"PDDL2.1: an extension to PDDL for expressing temporal planning domains", Fox M. and Long D., JAIR Vol. 20, 2003.

PDDL Example (i)

```
(:          action LOAD-TRUCK
  :parameters
 (?obj – obj ?truck – truck ?loc - location)

  :precondition
   (and                 (at ?truck ?loc)
                        (at ?obj ?loc))
  :effect
   (and              (not (at ?obj ?loc))
                     (in ?obj ?truck)))
```

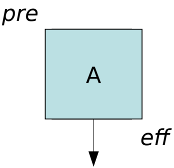
PDDL Example (i)

```
(:durative-action LOAD-TRUCK
  :parameters
 (?obj - obj ?truck - truck ?loc - location)
  :duration (= ?duration 2)
  :condition
   (and   (over all (at ?truck ?loc)
          (at start (at ?obj ?loc))
  :effect
   (and (at start (not (at ?obj ?loc))
          (at end (in ?obj ?truck)))
```

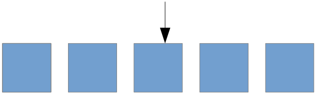
Beware of self-overlapping actions!

"Complexity of concurrent temporal planning", Rintanen J., ICAPS 2007

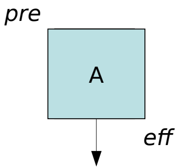
Durative Actions?



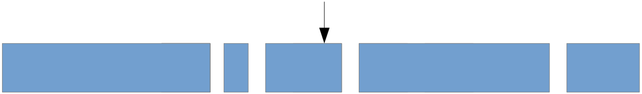
Classical Planner



Durative Actions?



Classical Planner



Temporal Planners in IPC 2003

Planner	Solved	Attempted	Success Ratio	Tracks entered
FF	237 (+70)	284 (+76)	83% (85%)	S, N, HN
LPG	372	428	87%	S, N, HN, ST, T
MIPS	331	508	65%	S, N, HN, ST, T, C
SHOP2	899	904	99%	S, N, HN, ST, T, C
Sapa	80	122	66%	T, C
SemSyn	11	144	8%	S, N
Simplanner	91	122	75%	S
Stella	50	102	49%	S
TALPlanner	610	610	100%	S, ST, T
TLPlan	894	894	100%	S, N, HN, ST, T, C
TP4	26	204	13%	N, ST, T, C
TPSYS	14	120	12%	ST, T
VHPOP	122	224	54%	S, ST

Winner, Fully Automated: LPG, solved more problems because it also handled temporal domains.

PDDL Example (ii)

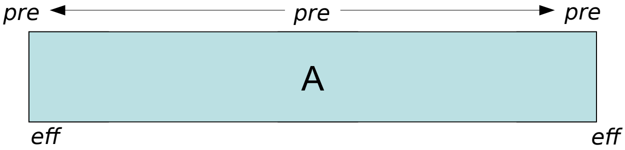
```
(:durative-action open-barrier
  :parameters
 (?loc - location ?p - person)
  :duration (= ?duration 1)
  :condition
   (and   (at start (at ?loc ?p)))
  :effect
   (and (at start (barrier-open ?loc))
        (at end (not (barrier-open ?loc))))
```

PDDL Example (ii)


```
(:durative-action op
  :parameters
 (?loc - location ?p
  :duration (= ?dura
  :condition
   (and   (at start
  :effect
   (and (at start (barrier-open ?loc))
        (at end (not (barrier-open ?loc))))
```

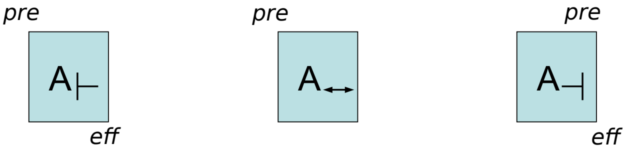
Durative Actions in LPGP

(Fox and Long, ICAPS 2003)



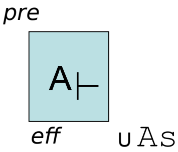
Durative Actions in LPGP

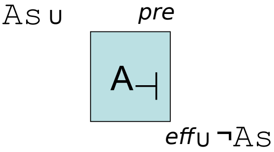
(Fox and Long, ICAPS 2003)



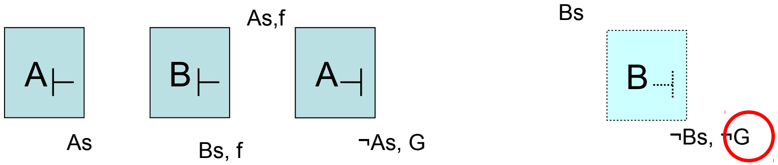
Durative Actions in LPGP

(Fox and Long, ICAPS 2003)





Planning with Snap Actions (i)

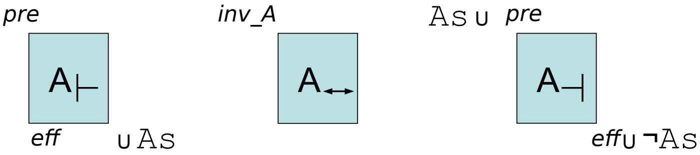


Challenge 1: What if B⊣ interferes with the goal?

- PDDL 2.1 semantics: **no actions can be executing in a goal state.**

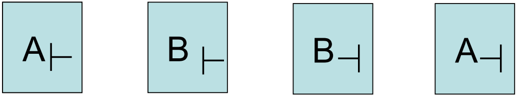
- **Solution:** add ¬As, ¬Bs, ¬Cs.... to the goal
  - (Or make this implicit in a temporal planner.)

Planning with Snap Actions (ii)



- Challenge 2: what about **over all** conditions?
  - If A is executing, inv_A must hold.
- **Solution:**
  - In every state where As is true: inv_A must also be true
  - Or: `(imply (As) inv_A)`
  - Violating an invariant then leads to a **dead-end**.

Planning with Snap Actions (iii)



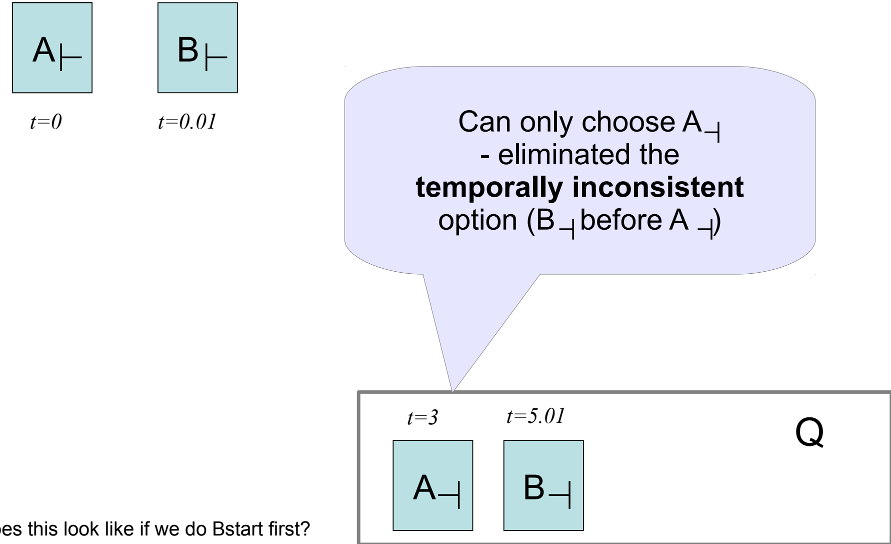
- Challenge 3: **where did the durations go?**
  - More generally, what are the temporal constraints?
  - **Logically sound ≠ temporally sound.**

Option 1: Decision Epoch Planning

- Search with **time-stamped states** and a **priority queue** of pending end snap-actions.
  - See e.g. Temporal Fast Downward (Eyerich, Mattmüller and Röger); Sapa (Do and Kambhampati).
- In a state S, at time t and with queue Q, either:
  - Apply a start snap-action $A_{\vdash}$ (at time t)
    - Insert $A_{\dashv}$ into Q at time $(t + dur(A))$
    - $S'.t = S.t + \varepsilon$
  - Remove and apply the first end snap-action from Q.
    - $S'.t$ set to the scheduled time of this, plus ε

"Using the Context-enhanced Additive Heuristic for Temporal and Numeric Planning." Eyerich P., Mattmüller R. and Röger G., ICAPS 2009
"Sapa: A Scalable Multi-Objective Metric Temporal Planner", Do M. and Kambhampati S., JAIR 2003.

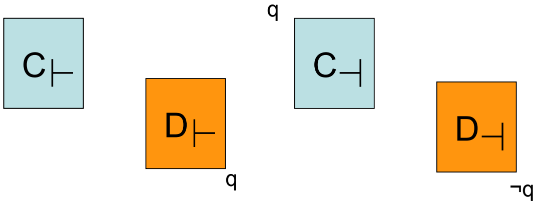
Running through our example...



What does this look like if we do Bstart first?

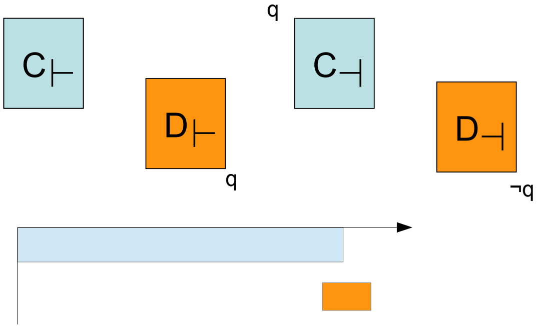
Decision Epoch Planning: The snag

- Must **fix start- and end-timestamps** at the point when the action is started.
  - Used for the priority queue
- Can we always do this?



Decision Epoch Planning: The snag

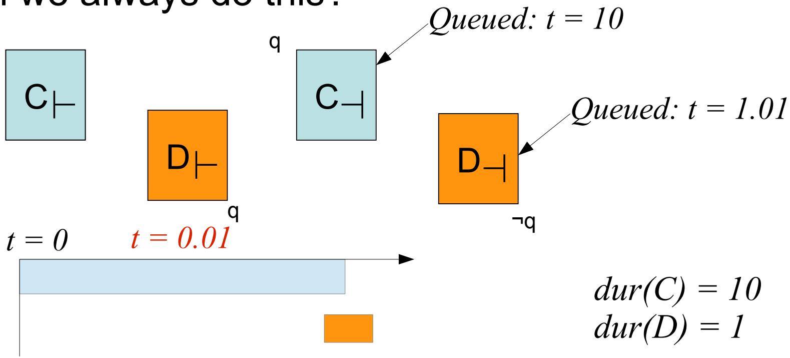
- Must **fix start- and end-timestamps** at the point when the action is started.
  - Used for the priority queue
- Can we always do this?



dur(C) = 10
dur(D) = 1

Decision Epoch Planning: The snag

- Must **fix start- and end-timestamps** at the point when the action is started.
  - Used for the priority queue
- Can we always do this?



IPC 2004 Planners

	ADL	DP	Numbers	Durations	TL
CRIKEY	-	-	+	+	-
FAP	-	-	-	-	-
FD, FDD	+	+	-	-	-
LPG-TD	+	+	+	+	+
Macro-FF	+	-	-	-	-
Marvin	+	+	-	-	-
Optop	+	+	+	+	+
P-MEP	+	-	+	+	+
Roadmapper	-	-	-	-	-
SGPlan	+	+	+	+	+
Tilsapa	-	-	+	+	+
YAHSP	-	-	-	-	-

"The Deterministic Part of IPC-4: An Overview" Hoffmann J. and Edelkamp S., JAIR Special Issue on the 4th International Planning Competition 2005.

Simple Temporal Networks: VHPOP and CRIKEY!

"Temporal Constraint Networks", Dechter, Meiri and Pearl, Artificial Intelligence, 1991
"VHPOP: Versatile heuristic partial order planner" Younes H. and Simmons R., JAIR Vol 20, 2003.
"Planning with Problems Requiring Temporal Coordination." A. I. Coles, M. Fox, D. Long, and A. J. Smith. AAAI 08.
"Managing concurrency in temporal planning using planner-scheduler interaction." A. I. Coles, M. Fox, K. Halsey, D. Long, and A. J. Smith. Artificial Intelligence, 173 (1), 2009.

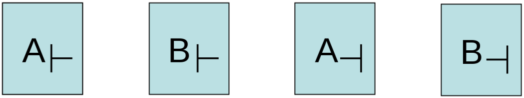
Option 2: a Simple Temporal Problem

- All our constraints are of the form:
  - $\varepsilon \leq t(i+1) - t(i)$ *(c.f. sequence constraints)*
  - $dur_{min}(A) \leq t(A_{\dashv}) - t(A_{\vdash}) \leq dur_{max}(A)$
- Or, more generally, $lb \leq t(j) - t(i) \leq ub$
  - Is a **Simple Temporal Problem**
  - “Temporal Constraint Networks”, Dechter, Meiri and Pearl, AIJ, 1991
- Good news – is **polynomial**
  - Bad news – in planning, we need to solve it a lot....

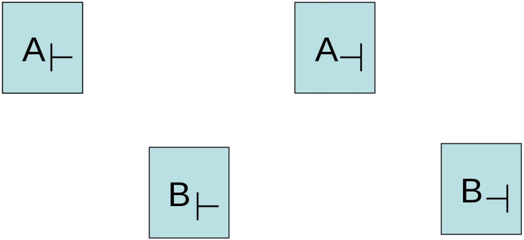
Simple Temporal Networks

- Can map STPs to an equivalent digraph:
  - One vertex per time-point (and one for 'time zero');
  - For $lb \leq t(j) - t(i) \leq ub$:
    - An edge (i $\rightarrow$ j) with weight *ub*.
    - An edge (j $\rightarrow$ i), with weight *-lb*
      - (c.f. $lb \leq t(j) - t(i) \quad \rightarrow \quad t(j) - t(i) \leq -lb$)

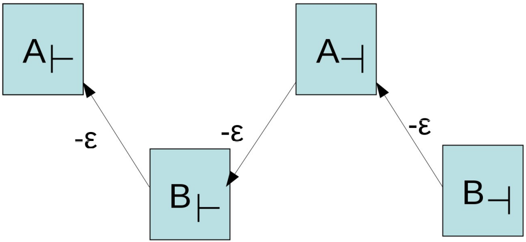
STN Example



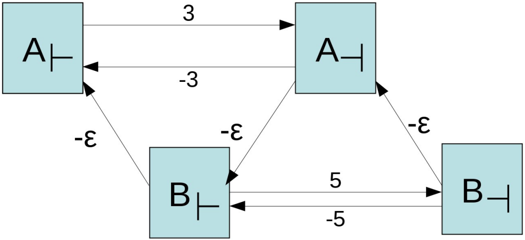
STN Example



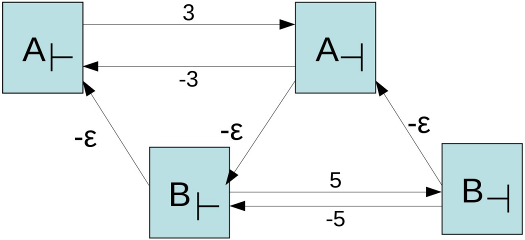
STN Example



STN Example



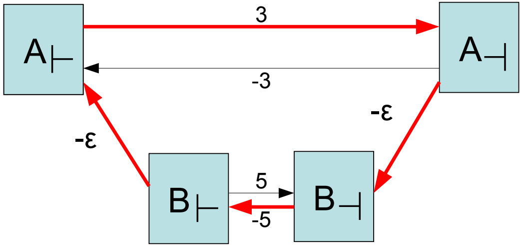
STN Example




```
0.00: (A) [3]
0.01: (B) [5]
```

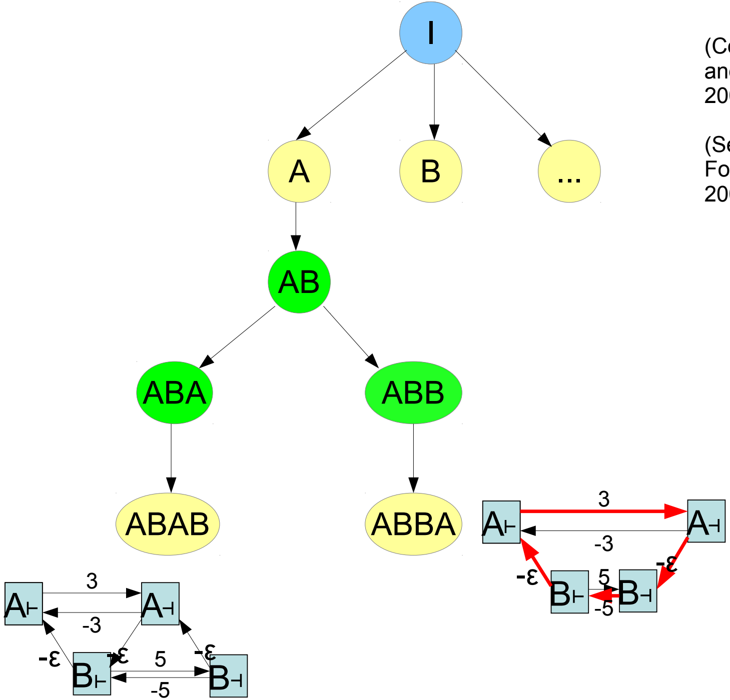
Simple Temporal Networks (ii)

- Solve the shortest path problem (e.g. using Bellman-Ford) from/to zero
  - dist(0,j)=x → maximum timestamp of j = x
  - dist(j,0)=y → minimum timestamp of j = -y
- If we find a **negative cycle** then the temporal constraints are inconsistent:



CRIKEY! (3)

"Planning with Problems Requiring Temporal Coordination." A. I. Coles, M. Fox, D. Long, and A. J. Smith. AAAI 08.
"Managing concurrency in temporal planning using planner-scheduler interaction." A. I. Coles, M. Fox, K. Halsey, D. Long, and A. J. Smith. Artificial Intelligence. 173 (1). 2009.



(Coles, Fox, Long and Smith, AAAI 2008);

(See also Halsey, Fox and Long, ECAI 2004)

Other fiddly details

- **The closed list** is a headache;
- Classical planning:
  - Discard states that are the same (in terms of facts, or same/worse cost) as states already seen.
- Temporal planning:
  - **Facts don't tell us everything** – due to the temporal constraints, the plan steps matter too.
  - ...as does their order – plans with different **permutations** of actions are interestingly different

"Have I Been Here Before? State Memoization in Temporal Planning", A.J. Coles & A.I. Coles ICAPS 2016.
Paper: https://www.aaai.org/ocs/index.php/ICAPS/ICAPS16/paper/view/13187
Talk Video: https://youtu.be/AwL1A25tjYo?list=PLj-ZdQ5rfSEpnsOfJeG7UfheAuZ42tEOM&t=928

IPC 2004: Results

	D		D+TL		D+NV		D+TL+NV	
Number	302		116		272		136	
CRIKEY	47	66	—		98	55	—	
FAP	—		—		—		—	
FD	—		—		—		—	
FDD	—		—		—		—	
LPG-3	45	62	—		56	50	—	
LPG-TD	76	62	63	100	96	50	87	100
Macro-FF	—		—		—		—	
Marvin	—		—		—		—	
Optop			8	43	—		—	
P-MEP	24	45	24	43	13	32		
Roadmapper	—		—		—		—	
SGPlan	75	90	78	74	85	100	74	100
Tilsapa			10	69			62	63
YAHSP	—		—		—		—	

Right: % of instances attempted, left % of these solved

D: Durative Actions

NV: Numeric Variables

TL: Timed Initial Literals

Note: Change of rules, temporal track now separate. LPG3: last year's winner.
Metric used: scalability (problems solved)

"The Deterministic Part of IPC-4: An Overview" Hoffmann J. and Edelkamp S., JAIR Special Issue on the 4th International Planning Competition 2005.

We will focus on generic techniques

A tuned planner

if domain name begins with "PS" and part after first letter is "SR":
  use algorithm 100
else if there are 5 actions, all with 3 args, and 12 non-ground predicates:
  use algorithm −1000
else if all predicates ground and 10th/11th domain name letters "PA":
  use algorithm −1004
else if there are 11 actions and action name lengths range from 5 to 28:
  use algorithm 107
. . .

PDDL 2.2: Timed Initial Literals

- Introduced in PDDL 2.2 (IPC 2004);
- Allow us to model facts that become true, or false, at a specific time.
- Can use them to model deadlines or time windows.
- Cannot be done directly, but we can achieve this by adding more facts to the domain.

Modelling Deadlines using TILs

- Make sure the action achieving the desired fact has a condition that ensures it takes place before the deadline (over all or at start/ end).
- Make that fact true in the initial state.
- And a TIL to delete it at the deadline.
- Note that we could have multiple deadlines for different objects.

```
(:durative-action unload-truck
  :parameters (?p - obj ?t- truck ?l-
location)
  :duration (= ?duration 2)
  :condition  (and (over all (at ?t ?l))
                   (at start (in ?p?t)))
                   (at end (can-deliver ?p)))
  :effect   (and (at start (not (in ?p ?t)))
               (at end (at ?p ?l))))
Init:
(can-deliver package1)
(at 9 (not (can-deliver package1)))
(can-deliver package2)
(at 11 (not (can-deliver package2)))
```

Modelling Time Windows Using TILs

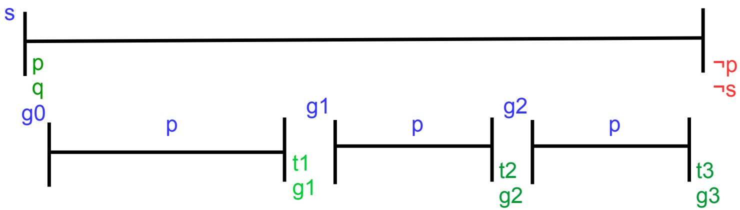
- Make sure the action achieving the desired fact has a condition that ensures it takes place during the window (over all or at start/end). POPF/OPTIC will generally work better if you use over all where possible.
- Have a TIL to add that fact at the starting point for the window.
- And one to delete it when the window ends.
- Note that we could have multiple windows for the same fact by adding further TILs to the initial state.

```
(:durative-action bus-route
  :parameters (?d – driver ?r – route ?b – bus
                    ?from ?to – loc)
  :duration (= ?duration (route-duration ?r))
  :condition (and   (at start (route ?r ?from ?to))
                  (at start (at ?d ?from))
                  (at start (at ?b ?from))
                  (over all (working ?d))
                  (at end (due ?r)))
 :effect (and   (at start (not (at ?d ?from)))
               (at start (not (at ?b ?from)))
               (at end (at ?d ?to))
               (at end (at ?b ?to))
               (at end (done ?r))
)
init:
(at 3.75 (due route2))
(at 4 (not (due route2)))
```

Reasoning with TILs

- TIL Sapa
  - Before search starts add all TILs to the event queue at the time they must occur.
- CRIKEY! (3)
  - Consider TILs as actions that can be applied in search, check temporal consistency as applied.
- LPG
  - Local search approach: when a TIL precondition is not satisfied either:
    - Remove the action;
    - Delay the action until after the TIL is true;
    - Remove earlier actions so that the action can occur sooner.

Compiled TIL Domains
Pipes, Airport, Satellite, UMTS



- q is an invariant condition of all ‘real’ actions in the domain, gn becomes a goal.
- Introduces required concurrency, making temporally interesting domains;
- Cannot be handled by planners using action compression (although the original TIL models can).
- Compilation makes problems much harder to solve.

IPC 2006

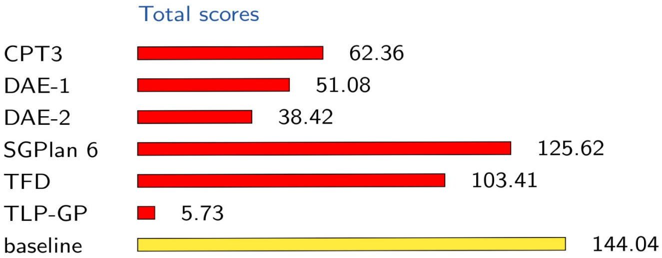
Gerevini, Dimopolous, Haslum and Saetti

- Focus on Metrics measuring Plan Quality, not just coverage/speed: tracks again merged together (no separate temporal track), overall satisfycing track winner SGPlan.

IPPlan	STRIPS
Fast Downward	STRIPS
Hplan-P	STRIPS, Simple Preferences, Qualitative Preferences
MIPS-XXL	STRIPS, Simple Preferences, Qualitative Preferences, Time
Yochan-PS	STRIPS, Simple Preferences, Time
SGPlan	STRIPS, Simple Preferences, Qualitative Preferences, Time

- First (makespan) Optimal Temporal Planner in Competition: Winner CPT (Vidal & Tabary) works by compilation to constraint programming. No other competitors, subsequent years cancelled due to only having one participant.
- Temporal Preferences introduced, handled by MIPS-XXL (and SGPlan). Preference tracks also did not run after 2006.
- No required concurrency.

IPC 2008

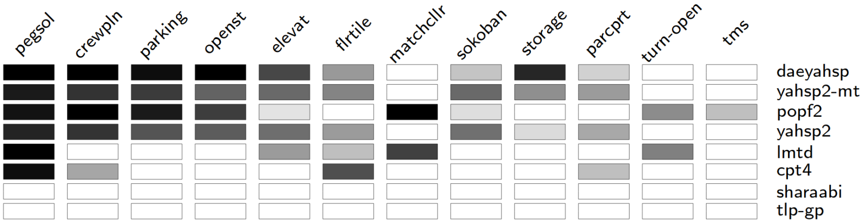


- ‘Baseline’ performed best – throw time away, run a classical planner. No temporally interesting domains, so this worked very well.
- SGPlan 6 was the best competitor – also ignored time
- TFD – Decision Epoch Planner
- DAE – decomposed by learning a goal agenda
- CPT – optimal temporal planning using CP
- TLP-GP – temporally expressive planner, based on regression in planning graphs

IPC 2011

- Return of some temporally interesting domains:
  - TMS (required concurrency bake during fire kiln)
  - Turn and Open (turn handle and open door)
  - Match Cellar (mend fuse whilst match is lit).

IPC 2011



- Winner: DAE, now with YAHSP – a forward-search planner with lookahead. Not temporally expressive, so no problems solved in matchcellar, turn-and-open and TMS.
- Joint runners up: YAHSP without DAE; and POPF – the only competitive planner to solve temporally expressive problems
- LMTD: prototype landmark heuristic with TFD
- Sharaabi: extension of SAPA to increase temporal expressivity

IPC 2014

- 10 domains, incl. 3 temporally interesting ones (from 2011).
- 5 Participants:
  - ITSAT: SAT-Based Temporally Expressive Planner.
  - tBURTON: Uses sub-goals and calls a sub-planner (TFD). Temporally Expressive if sub-planner is.
  - Temporal Fast Downward.
  - YAHSP3 and YAHSP3-MT (MT = multi-threaded)
  - DAE-YAHSP.

YAHSP3-MT	86.5/200	**1st**
Temporal-FD	79.2/200	**2nd**
YAHSP3	66.6/200	3rd

“The 2014 International Planning Competition: Progress and Trends” Vallati, M. and Chrpa, L. and Grzes, M. and McCluskey, T.L. and Roberts, M. and Sanner, S. AI Magazine, 2015.

IPC 2018

Domain/Planner	PopCorn	TemPorAl	TFLAP	CP4TP
Road-traffic	0	8.04	0	6.67
trucks	4.86	**8.99**	7.32	4.5
Map-analyzer	0	**9.05**	6.29	5.12
sokoban	0.76	**5.28**	4	4.05
airport	3	**9.98**	8.99	8.9
parking	4	6.17	3.65	2.85
quantum	4.38	**8.74**	7	7.82
cushing	1	0	2.98	2.22
floortile	0	**9.55**	2.9	1.59
Total	18	**65.8**	43.13	43.72

- First portfolios in the temporal track: TemPorAl and CP4TP. The former did not use a temporally expressive planner; the latter did (ITSAT), so could solve problems in the 'Cushing' domain.
- TFLAP – forward partial-order planner, with landmark and relaxed-plan heuristics. Competitive with CP4TP – a portfolio!
- PopCorn – a planner for domains with control parameters (not tested in the competition)

Recent Work/Challenges in Temporal Planning

- Much work in temporal planning is outwith PDDL2.1, e.g. timeline-based approaches (Frank, Chen, Smith, Cesta, Oddi, Fratini, ….)
- Reasoning efficiently with more interesting temporal constraints;
  - Relaxation heuristics for time windows (Allard et al); MTP (To et al); FAPE (Bit Monnot & Smith); Temporal Landmarks (Marzal et al; Wang et al); effective memoisation and metastates (Coles et al)
- RoboCup Logistics League Competition (robocup.org/leagues/17)
- Plan execution, including with temporal uncertainty (Chen et al)
- Hybrid Planning (e.g. PDDL+), interaction of time and numbers:
  - UPMurphi (Della Penna et al), DiNo (Piotrowski et al), PluReal (Bryce), OPTIC+ (Coles^2), SMTPlan+ (Cashmore et al), Kongming (Li & Williams).
- Applications work: Retirement Home Assistance, Space, Liner Shipping, Aerial Surveillance, Mining.