The LAMA Planner

Silvia Richter
Griffith University & NICTA, Australia

Joint work with Matthias Westphal
Albert-Ludwigs-Universität Freiburg

February 13, 2009

Outline

Outline

LAMA is a state-of-the-art heuristic search planner.

Won the satisficing track of the 6th International Planning Competition (IPC-2008).

Based on Fast Downward (Helmert & Richter), winner of the satisficing track at IPC-2004.

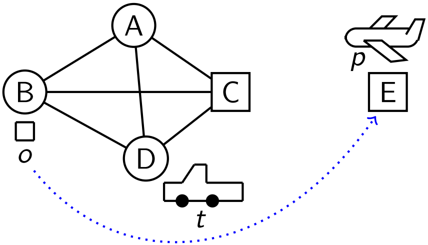
Core components we will discuss:

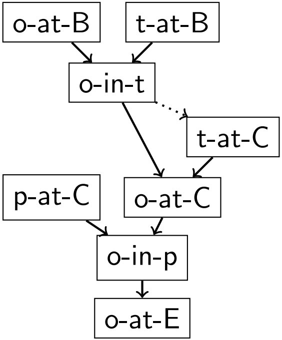
- Landmarks to direct search
- Multi-heuristic search
- Preferred Operators as further source of heuristic information
- Anytime search to try to make best use of given time

Outline

Landmarks

- Facts that must be true in every plan
  (Porteous & Cresswell 2002; Hoffmann et al. 2004)
- Intuitively helpful to direct seach
- Automatically found, incl. orderings



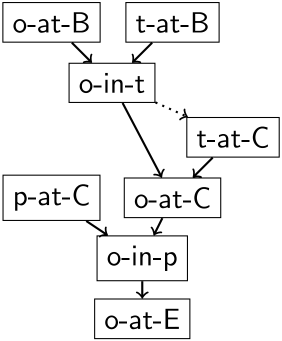


- Next: how to find landmarks, and how to use them

Landmark Generation I

Find landmarks by backchaining (Hoffmann et al. 2004)

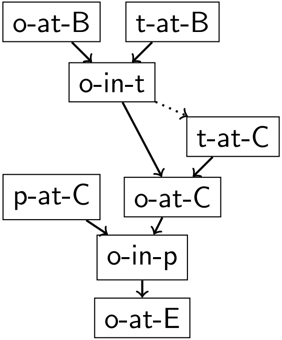
- Every goal is a landmark
- If B is landmark and all actions that achieve B have A as precondition, then A is a landmark
- Useful restriction: consider only the case where B is achieved for the first time ⇝ find more landmarks. (Why?)
- NP-hard to find all first achievers ⇝ over-approximation by building RPG without actions that add B. Any action applicable in this RPG can possibly be executed before B first becomes true.



Landmark Generation I

Disjunctive landmarks also possible, e.g., ($\text{o-in-p}_1 \vee \text{o-in-p}_2$):

- If B is landmark and all actions that (first) achieve B have either A or C as precondition, then A $\vee$ C is a landmark
- Generalises to any number of disjuncts, though usually restricted in practice
- Large number of possible disjunctive landmarks $\rightsquigarrow$ in practise, restrict set of facts, e.g. such that all facts must be instantiations of the same predicate.



Domain Transition Graphs (DTGs)

Find landmarks through DTGs (Richter et al. 2008)

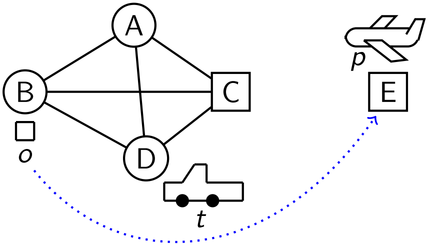
The domain transition graph of $v \in \mathcal{V}$ (DTG_v) represents how the value of v can change.

Given: a SAS^+ task $\langle \mathcal{V}, \mathcal{A}, s_0, s_\star \rangle$
DTG_v is a directed graph with nodes $\mathcal{D}_v$ that has arc $\langle d, d' \rangle$ iff

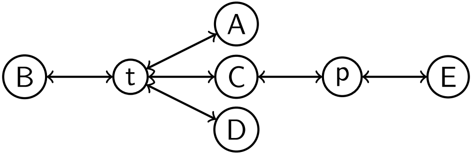
- $d \neq d'$, and
- $\exists$ action with $v \mapsto d'$ as effect, and either
  - $v \mapsto d$ as precondition, or
  - no precondition on v

$\rightsquigarrow$ DTG_v is the same as the graph of an atomic abstraction

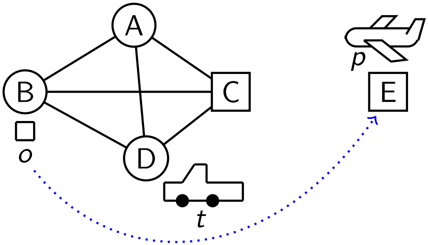
DTG Example



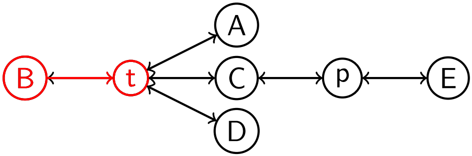
DTG for v_o:



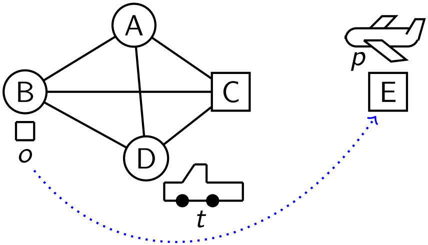
DTG Example



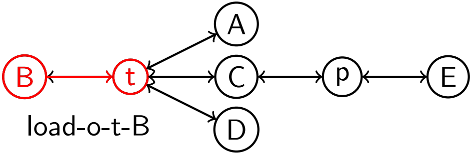
DTG for v_o:



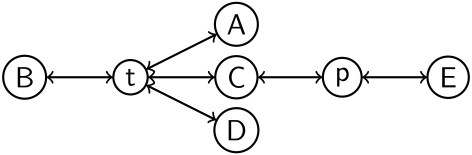
DTG Example



DTG for v_o:



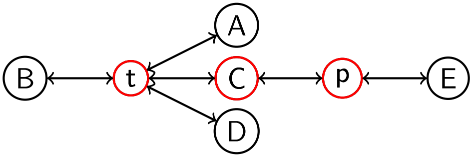
Landmark Generation II



- Find landmarks through DTGs: if
  - $s_0(v) = d_0$,
  - $v \mapsto d$ landmark, and
  - every path from d_0 to d passes through d',

then $v \mapsto d'$ landmark

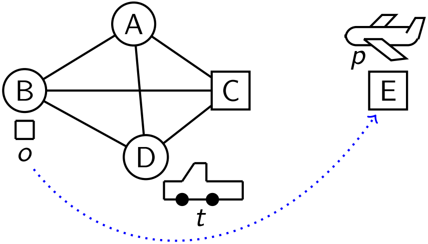
Landmark Generation II



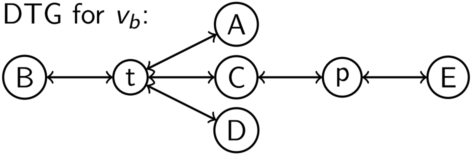
- Find landmarks through DTGs: if
  - $s_0(v) = d_0$,
  - $v \mapsto d$ landmark, and
  - every path from d_0 to d passes through d',

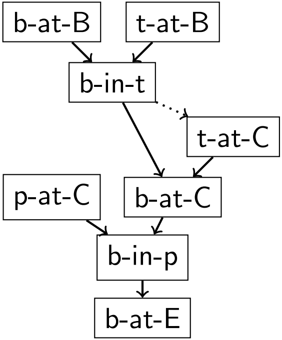
then $v \mapsto d'$ landmark

Landmark Generation II (ctd.)

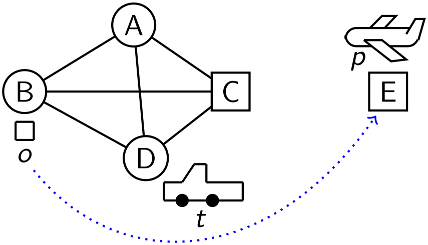


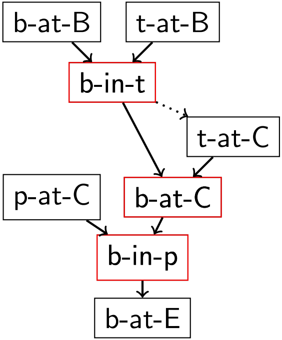
DTG for v_b:



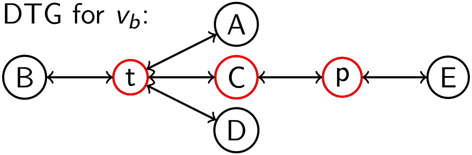


Landmark Generation II (ctd.)

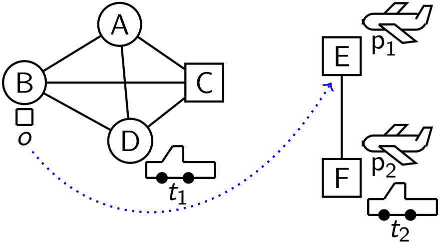


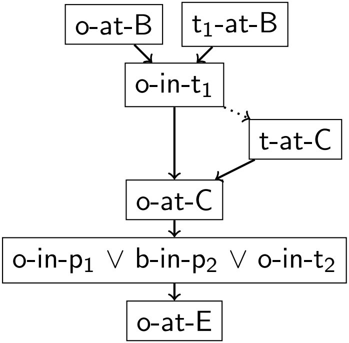


DTG for v_b:

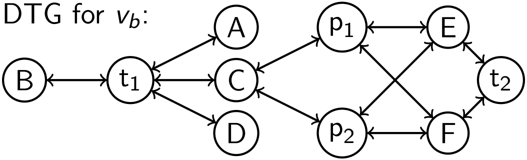


Landmark Generation II (ctd.)

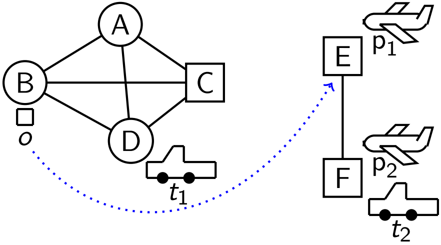


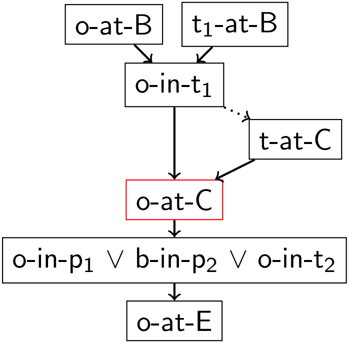


DTG for v_b:

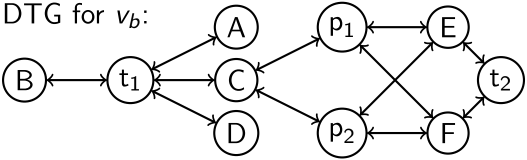


Landmark Generation II (ctd.)

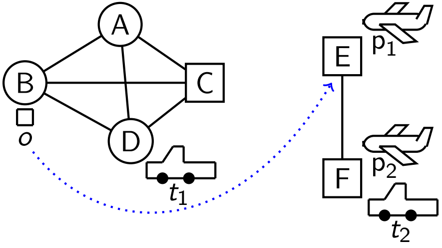




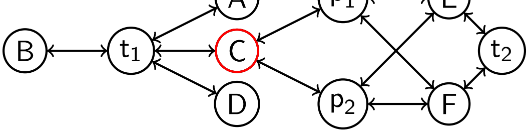
DTG for v_b:

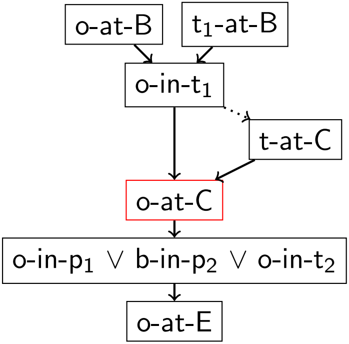


Landmark Generation II (ctd.)



DTG for v_b:

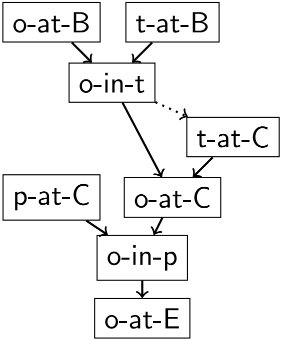




Landmark Orderings

Find orderings: given two landmarks A and B,

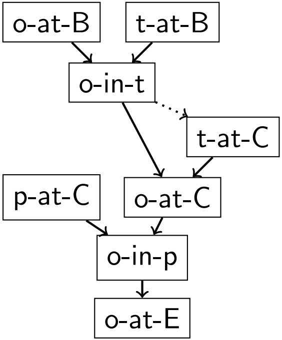
- if A was found by backchaining from B, then A can be ordered before B.
- if it is not possible to reach B before A in an RPG, then A can be ordered before B.
- Reasonable orderings: if we have to make B false in order to achieve A, then it "makes sense" to achieve A before B. These orderings may not always be correct, but can help in practise!



Using Landmarks I

Using landmarks in localised search approach

- Make the landmarks subgoals, then simply concatenate plans of subtasks (Hoffmann et al. 2004)
- Greatly speeds up search in many domains
- Any base planner possible for subtasks
- But: Incomplete (dead ends), and bad-quality plans



Using Landmarks II

Using landmarks in global search approach

- Heuristic = #landmarks that still need to be achieved (Richter et al. 2008)
- Takes landmark orderings into account: some landmarks may have to be achieved more than once if they are preconditions for other landmarks
- Pseudo-Heuristic because depends on path to state
- Can be combined with other heuristics through multi-heuristic BFS (more details later)

Landmarks in LAMA

- Backchaining and DTGs to find landmarks
- Also disjunctive landmarks
- Landmark heuristic with action costs incorporated:
  rather than counting the number of missing landmarks, sum over lower bounds on their costs

Outline

Multi-heuristic Search

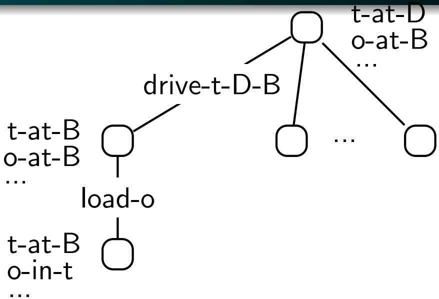
Several heuristics can be combined for better performance (Helmert 2006):

- In a heuristic search using open lists (e.g. BFS, A^*,...) use a separate open list for each heuristic
- Evaluate states with all heuristic functions and put in all corresponding open lists
- Select next state alternatingly from the various open lists

(Why should we not use a single open list?)

LAMA: combines landmark heuristic with cost-sensitive variant of FF-heuristic

Preferred Operators



- Second source of heuristic information
- Idea: prefer actions that are likely to improve heuristic value, and try these actions before others $\rightsquigarrow$ one-step look-ahead
- E. g., actions which are part of plan for simplified problem (Helmert 2006, Hoffmann & Nebel 2001)
- Often substantial performance improvement

Preferred Operators (ctd.)

- FF heuristic: preferred operators = "helpful actions": actions that are part of plan for relaxed task
- Landmark heuristic: preferred operators = landmark-achieving operators or operators in relaxed plan to nearest landmark
- LAMA: uses preferred operators for both FF heuristic and landmarks heuristic

Anytime Search

IPC-2008 requirement: find best possible plan within 30 minutes.
This suggests an anytime approach:

- Find a solution as quickly as possible
  (any solution is better than none).
  LAMA: greedy BFS.
- While there is still time, try to improve the solution.
  LAMA: series of weighted A^* searches with decreasing weights.
- Interesting finding: a series of independent runs of weighted A^* is better than one continued search (restarts overcome early mistakes)

Outline

The IPC 2008 tracks

Three track categories:

- sequential:
  - STRIPS + action costs
  - objective: minimize total cost (sum of action costs)
- temporal:
  - STRIPS + durative actions (+ numeric fluents)
  - objective: minimize total time (makespan)
- net benefit:
  - STRIPS + action costs + soft goals (+ numeric fluents)
  - objective: maximize net benefit
    (utility of achieved goals minus total cost)

Six tracks:

- for each category, a satisficing track and an optimization track
- awards given to winner and runner-up of each track
- one additional jury award

Sequential Satisficing Track

Number of competitors: 23 registered, 9 submitted
Competition setting for all satisficing tracks:

- 6–9 domains per track, 30 tasks each
- planners get score 0.0-1.0 for each solved task
- score is 1.00 for optimal or best known solutions
- otherwise score is

$$\frac{\text{cost of best known plan}}{\text{cost of generated plan}}$$

- highest aggregate score wins
- score only depends on plan quality
- runtime does not affect score
- limits per task: 30 minutes, 2 GB RAM

Participants in the sequential satisficing track

- C^3 (Miguel Ramírez, Nir Lipovetzky, Héctor Geffner): forward state space search with powerful structural pruning scheme based on inference along possible causal chains
- Divide-and-Evolve 1/2 (Jacques Bibai, Pierre Savéant, Marc Schoenauer, Vincent Vidal): subgoal decomposition, evolutionary algorithms + CPT
- DTGPlan (Ruoyun Huang, Yixin Chen, Weixiong Zhang): A^* search in hierarchically extended abstract search space, subplans generated from DTGs via causal analysis
- FF(h_a) and FF(h_{sa}) (Emil Keyder, Héctor Geffner): FF made cost-sensitive by using best supports of atoms (h_a) or propagating action sets rather than cost values (h_{sa}).

Participants in the sequential satisficing track (ctd.)

- LAMA (Silvia Richter, Matthias Westphal):
  Fast Downward with FF heuristic + landmark heuristic
  + iterated WA*
- Plan-A (Qiang Lv, Yixin Chen, Ruoyun Huang):
  DPLL-opt (DPLL with linear cost function optimization)
  + branch-and-bound pruning
- SGPlan 6 (Chih-Wei Hsu, Benjamin Wah)
  subproblem partitioning + Metric-FF
  + conflict resolution with Extended Saddle Point Condition
- baseline planner:
  throw away action costs, run FF

Sequential satisficing track: Results

C^3

DAE-1

DAE-2

DTGPlan

FF(h_a)

FF(h_{sa})

LAMA

Plan-A

SGPlan 6

(Upwards)

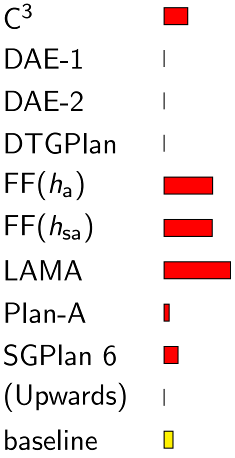
baseline

Sequential satisficing track: Results

Domain: Cyber security

C^3	10.65
DAE-1	0.00
DAE-2	0.00
DTGPlan	0.00
FF(h_a)	21.87
FF(h_{sa})	21.68
LAMA	29.92
Plan-A	2.27
SGPlan 6	6.27
(Upwards)	0.00
baseline	4.00

Sequential satisficing track: Results

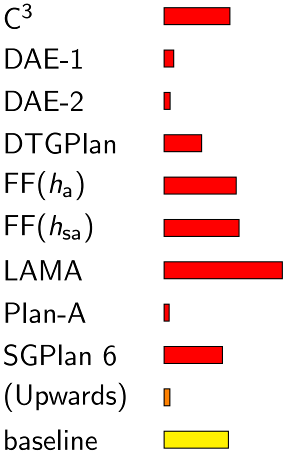


Sequential satisficing track: Results

Domain: Elevators

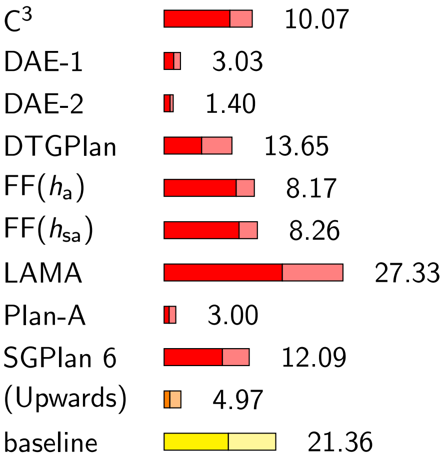
C^3	19.00
DAE-1	4.34
DAE-2	2.70
DTGPlan	16.92
FF(h_a)	10.62
FF(h_{sa})	12.09
LAMA	23.35
Plan-A	
SGPlan 6	20.00
(Upwards)	2.60
baseline	24.96

Sequential satisficing track: Results

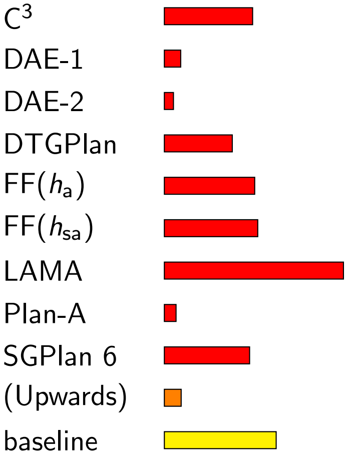


Sequential satisficing track: Results

Domain: Openstacks

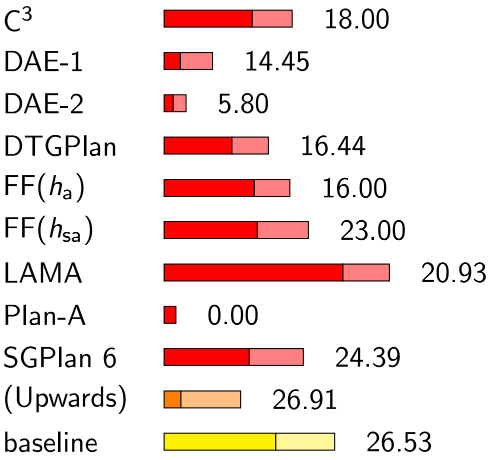


Sequential satisficing track: Results

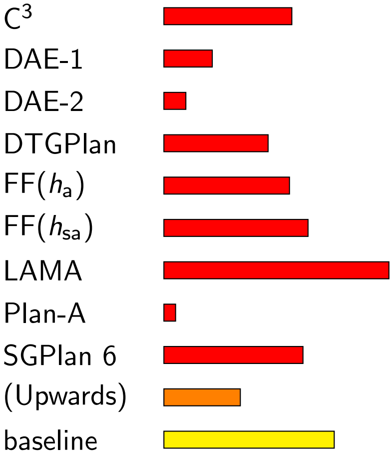


Sequential satisficing track: Results

Domain: ParcPrinter

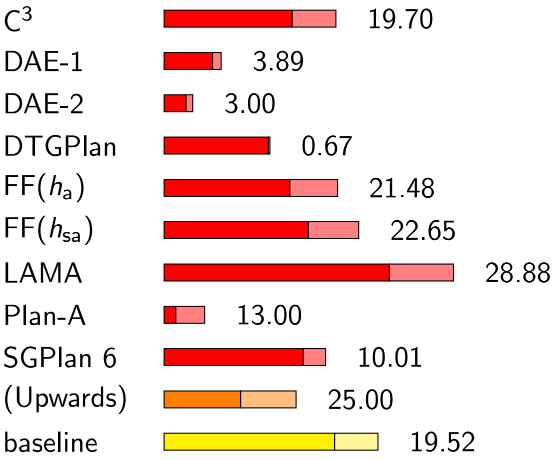


Sequential satisficing track: Results

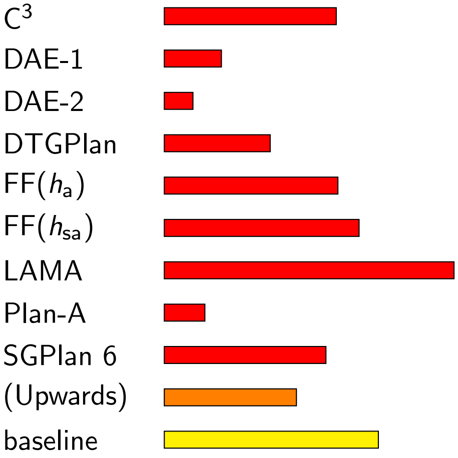


Sequential satisficing track: Results

Domain: Peg Solitaire

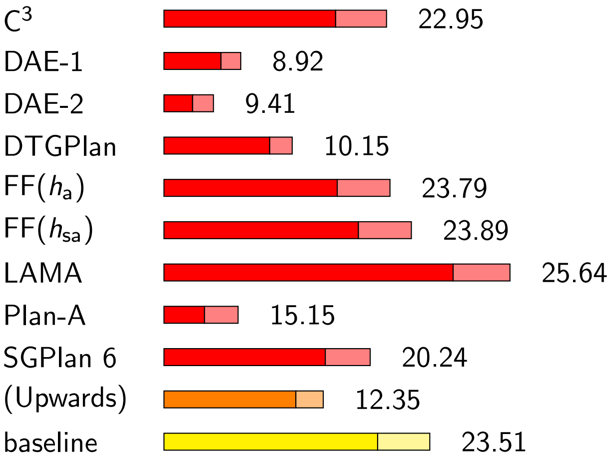


Sequential satisficing track: Results

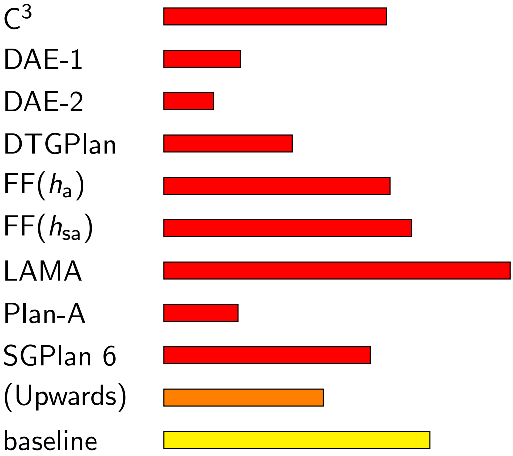


Sequential satisficing track: Results

Domain: Scanalyzer-3D

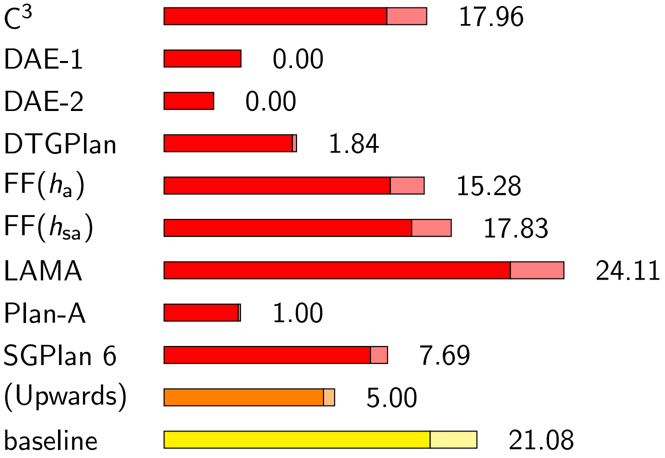


Sequential satisficing track: Results

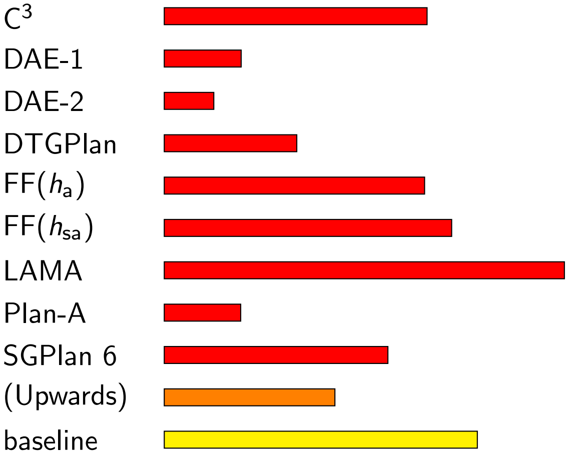


Sequential satisficing track: Results

Domain: Sokoban

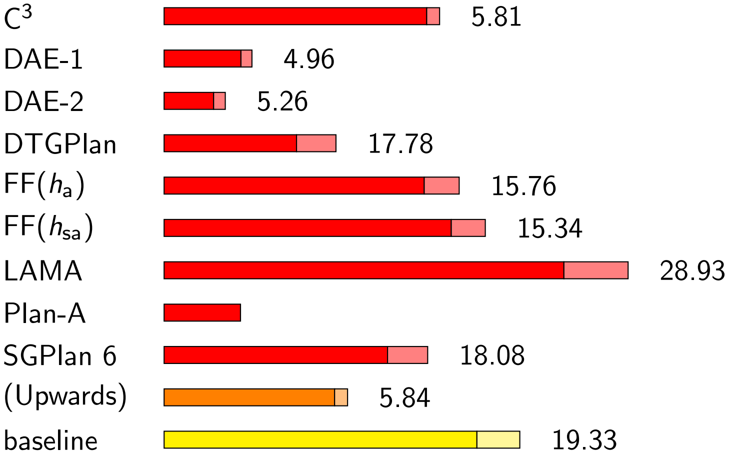


Sequential satisficing track: Results

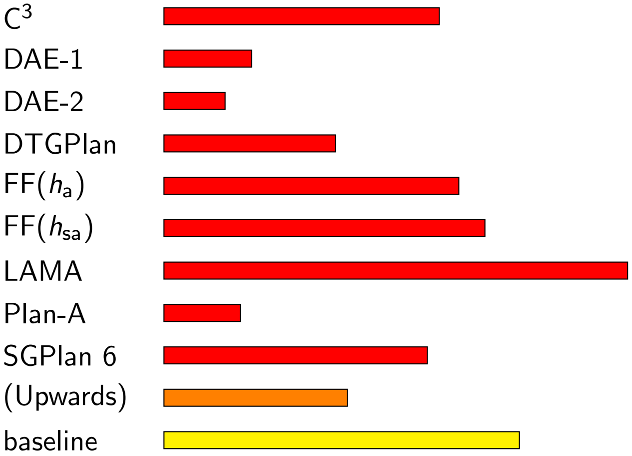


Sequential satisficing track: Results

Domain: Transport

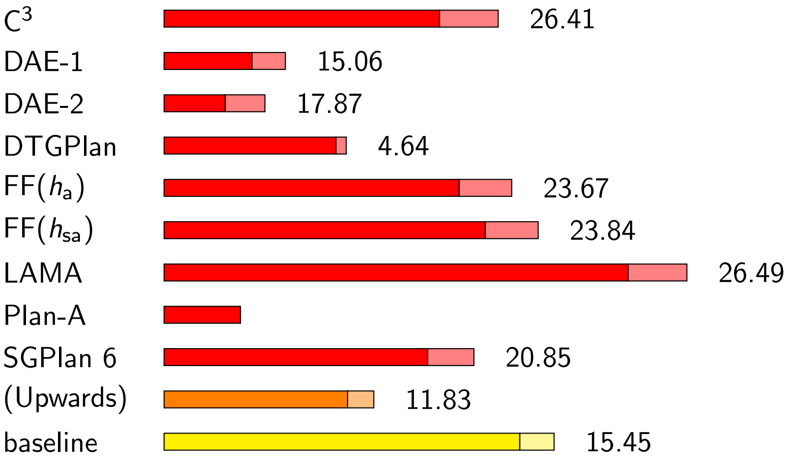


Sequential satisficing track: Results

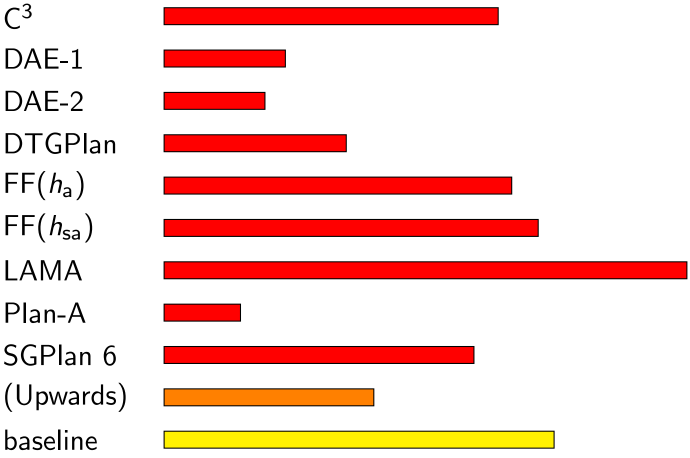


Sequential satisficing track: Results

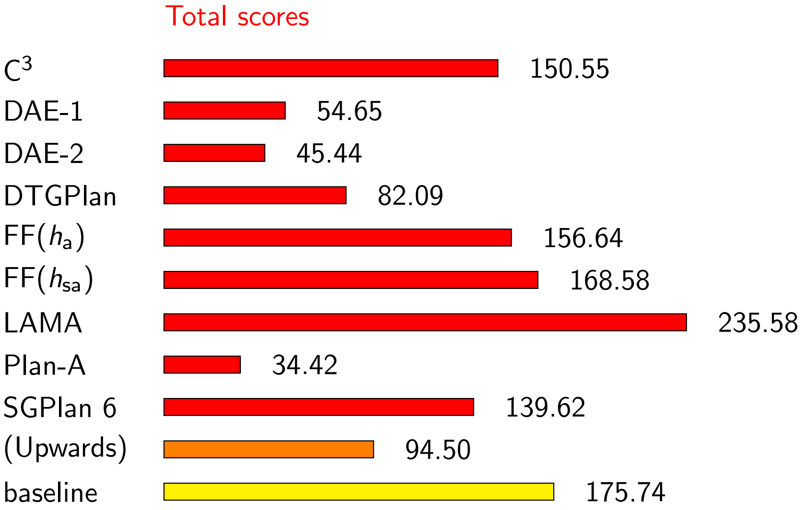
Domain: Woodworking



Sequential satisficing track: Results



Sequential satisficing track: Results



Sequential satisficing track: Awards

Winner

LAMA by Silvia Richter and Matthias Westphal

Runner-up

FF(h_{sa}) by Emil Keyder and Héctor Geffner

Jury Award

C^3 by Miguel Ramírez, Nir Lipovetzky and Héctor Geffner

References

- Jörg Hoffmann, Julie Porteous and Laura Sebastia.
  Ordered landmarks in planning.
  JAIR 22, pp. 215–278, 2004.
  Introduces landmarks and the backchaining generation method. (Longer version of a 2001 article by the same authors.)
- Julie Porteous and Stephen Cresswell.
  Extending landmarks analysis to reason about resources and repetition.
  Proc. PLANSIG 02, pp. 45–54, 2002.
  Introduces the "possibly before" approximation using RPGs for first achievers of landmarks.

References (ctd.)

- Silvia Richter, Malte Helmert and Matthias Westphal.
  Landmarks revisited.
  Proc. AAAI 2008, pp. 975–982, 2008.
  Describes DTG method for finding landmarks and the landmark heuristic.
- Malte Helmert.
  The Fast Downward planning system.
  JAIR 26, pp. 191–246, 2006.
  Describes the architecture of Fast Downward and LAMA.

References (ctd.)

- Jörg Hoffmann and Bernhard Nebel.
  The FF planning system: Fast plan generation through heuristic search.
  JAIR 14, pp. 253–302, 2001
  Describes the FF heuristic of which LAMA uses a cost-sensitive variant.