The LAMA Planner: Guiding Cost-Based Anytime Planning with Landmarks

Silvia Richter, Matthias Westphal
2009

Introduction

- HSP planning system (2001)
- FF planning system (2001)
- Fast Downward (2006)
- LAMA
  - Anytime search
  - Action costs
  - Landmarks

Outline

- Planning Domain
- System Architecture
- Anytime search
- Landmarks
- Landmark heuristics
- Experiments

Finite domain planning task

- Planning task in finite-domain representation: $\langle V, s_0, s_*, O, C \rangle$
  - V: set of state variables, with finite domain D
  - s_0: total variable assignment (initial state)
  - s_*: partial variable assignment (goal states)
  - O: operators <*pre*, *eff*>
  - C: action cost function

System Architecture

- Overall Structure of Fast Downward
- Three components:
  - Translation module
  - Knowledge compilation module
  - Search module

Search module

- Two algorithms:
  - Greedy best-first search
  - Weighted A*

- Search enhancements:
  - Deferred heuristic evaluation
  - Multi-queue search
  - Preferred operators

Roadmap for the rest of the talk

- Anytime search
- Landmarks

Weighted A*

- **The algorithm**
  - A* with weighted heuristic: $f(s) = g(s) + w * h(s)$
  - Weight $w \geq 0$ is input parameter
- **Properties**
  - w = 0 uniform cost search
  - w = 1 A*
  - w → ∞ GBFS

Anytime search

- **Objective**: find the cheapest solution within a given time limit

- **Approach**: Anytime search
  - First step: find some solution (can have arbitrary costs)
  - Second step: search for progressively cheaper solutions until the time limit is exceeded

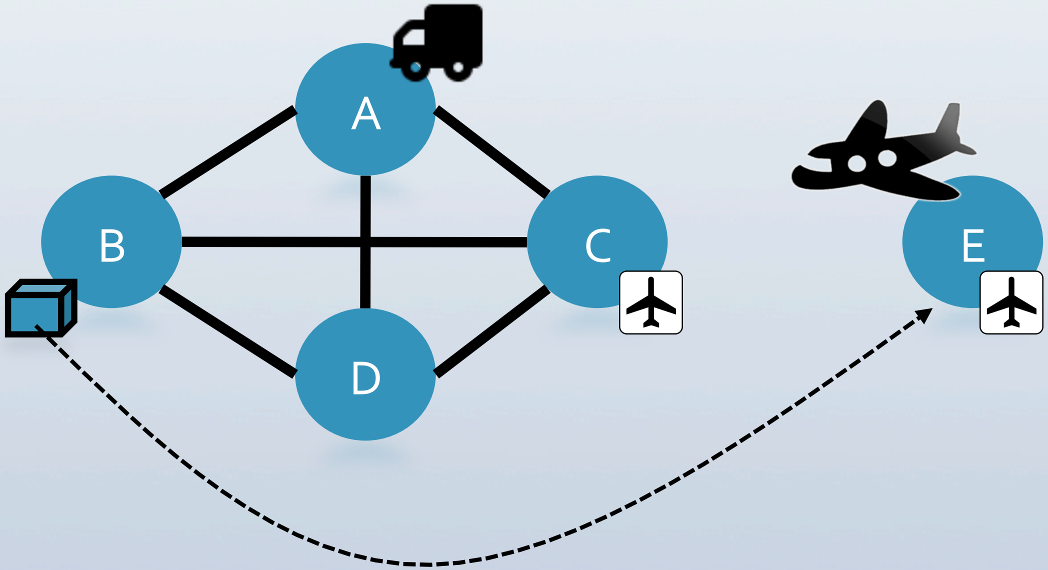
- **Implementation** in LAMA
  - First step: run GBFS
  - Run Weighted A* with decreasing weights (less greedy)
  - In each iteration, start **new** run of weighted A* (discard the open list!)

Roadmap for the rest of the talk

- Anytime search
- Landmarks

Landmarks

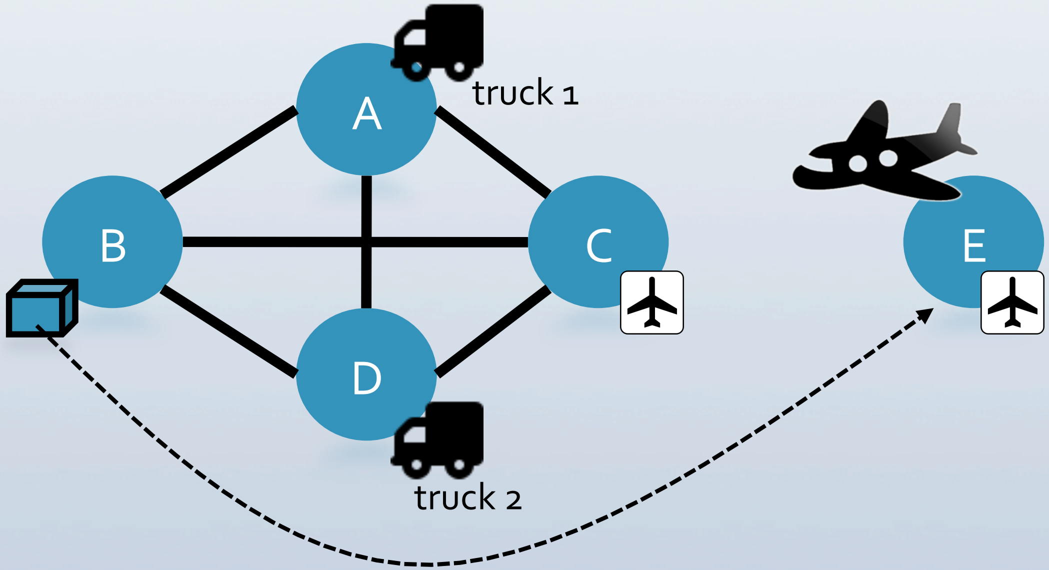
- Partial variable assignement
- Subgoals that must be achieved in every plan
- Partition taks into subproblems



- **Landmarks:**
  - Box is in truck
  - Box is at C
  - Box is in plane

Landmarks

- Partial variable assignement
- Subgoals that must be achieved in every plan
- Partition taks into subproblems



- **Landmarks:**
  - Box is in truck 1 or in truck 2
  - Box is at C
  - Box is in plane

disjunctive landmarks

Ordering

➔ Which landmarks to be achieved next

- Natural ordering, $\varphi \rightarrow \psi$

  if in each plan where ψ is true at time i**,** φ is true at time j<i

- Necessary ordering, $\varphi \rightarrow_n \psi$

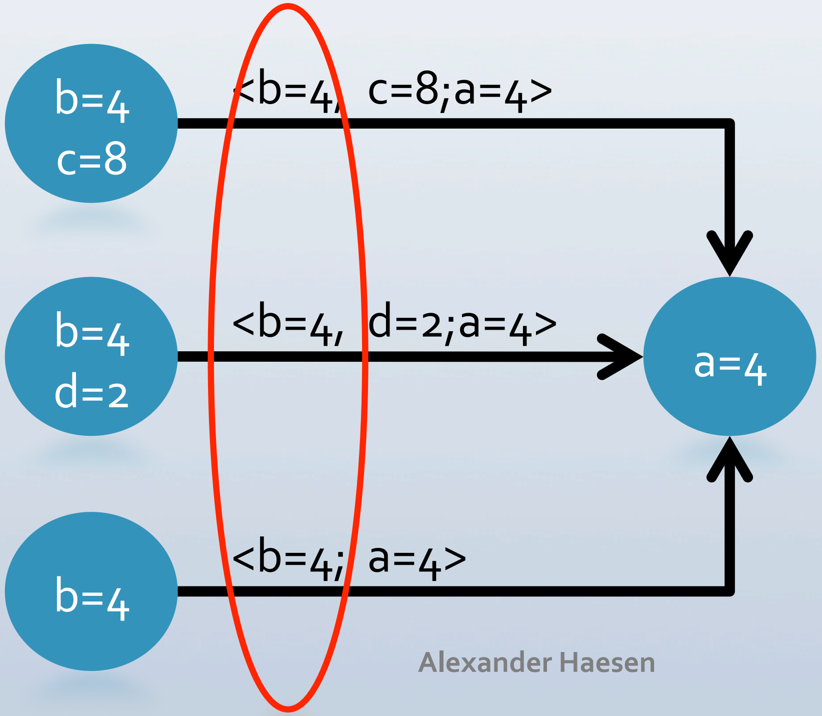
  If in each plan where ψ is added at time i, φ is true at i-1

- Greedy-necessary ordering,$\varphi \rightarrow_{gn} \psi$

  If in each plan where ψis **first** added at time i, φ is true at time i-1

Extracting Landmarks: Back-chaining

- Definitions:
  - First achiever
  - Shared precondition



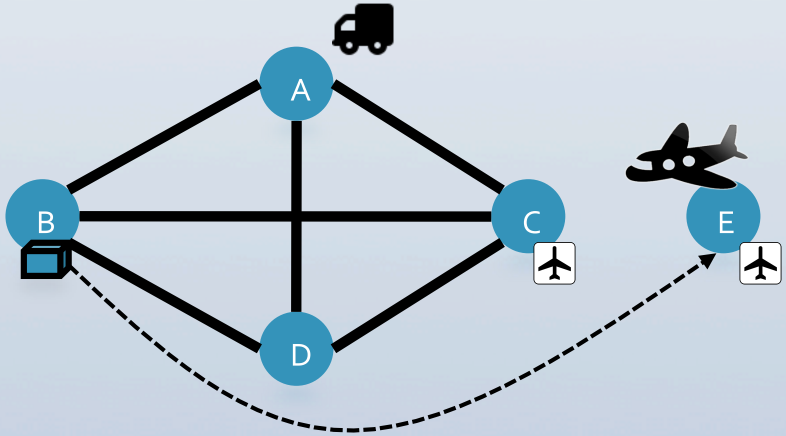
First achievers

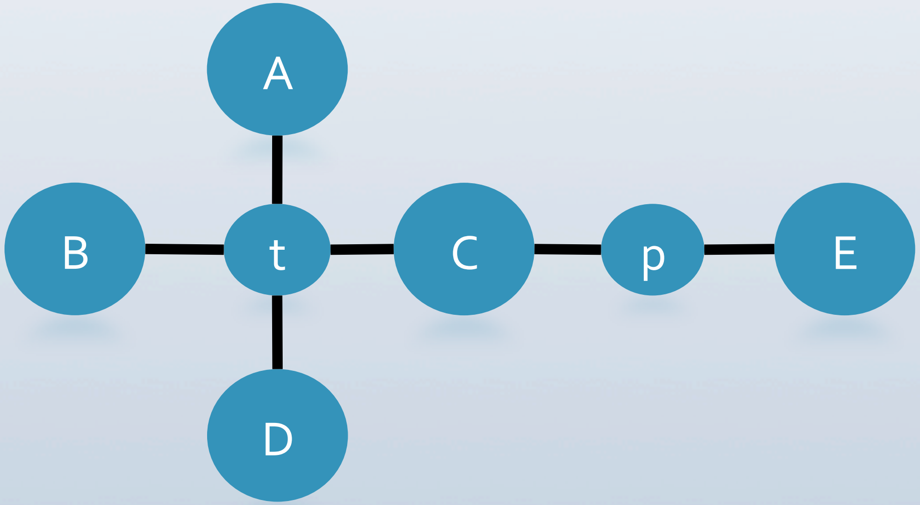
- Find first achievers of ψ is PSPACE-hard
- **Alternative**: over-approximation
  - Using restricted relaxed planning graphs
  - Leave out any operator that would make ψ true
  - When graph levels out, last set of facts is an over-approximation of the facts that could make ψ true

→ Any operator from this set that achieves ψ is a *possible first achiever* of ψ

Landmarks via Domain Transition Graph

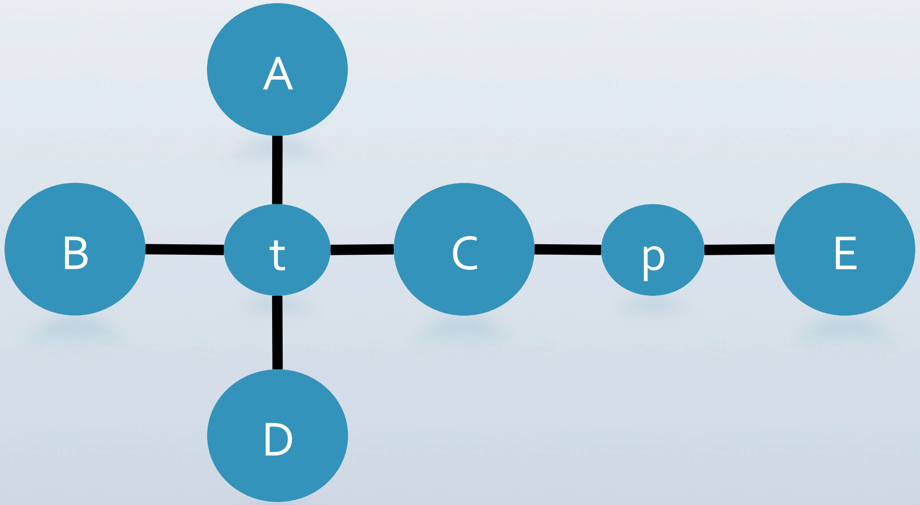
- Represents the different states that a variable can be assigned to
- Contains an arc between *d* and *d*' if there is an operator that changes *d* to *d*'





Landmarks via Domain Transition Graph

- If a node occurs on *every* path from s_0 to a *landmark value l*, it corresponds to a landmark *l*'



Landmark heuristic

- Estimation of goal distance:

$$L(s,\pi) \coloneqq (L\backslash\mathrm{Accepted}(s,\pi) \cup ReqAgain(s,\pi))$$

  - Accepted: Landmarks that have been achieved so far
  - ReqAgain: landmarks that have to be achieved again

- Intuition: Assigns higher values to states if fewer landmarks have already been achieved

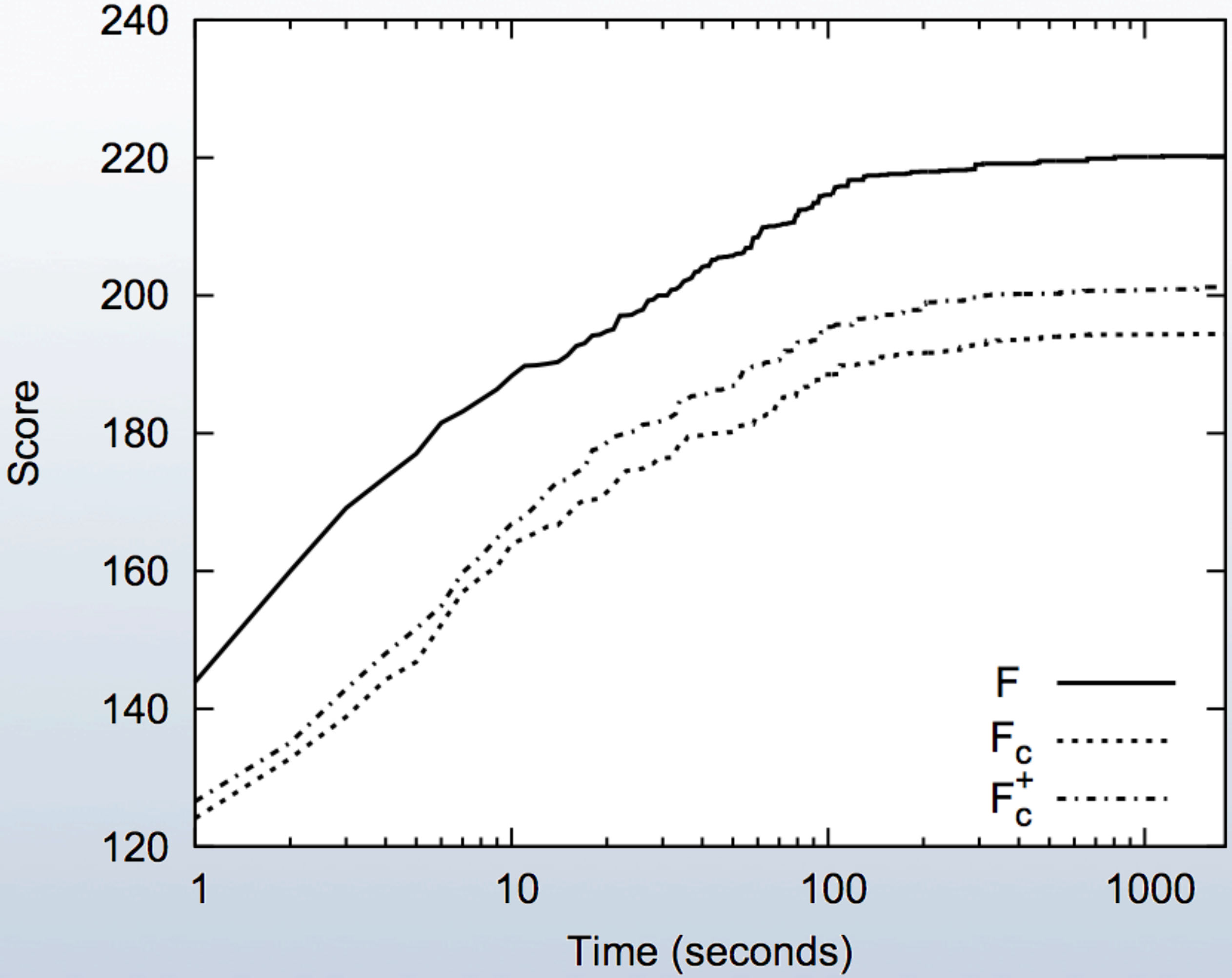
- Heuristic value:

$$h(s,\pi) \ \coloneqq \ |L(s,\pi)|$$

Experiments

		IPC Planner			Slowed Lama	
Domain	Base	C3	FF(h_a)	LAMA	x10	x100
Cyber Securtiy	4	9	20	28	27	26
Elevators	21	16	9	20	20	17
Openstacks	21	10	8	27	27	26
PARC Printer	27	18	16	21	19	12
Peg Solitaire	20	20	21	29	29	26
Scanalyzer	24	23	24	26	25	22
Sokoban	21	18	15	24	22	15
Transport	18	6	15	27	25	21
Woodworking	14	24	22	25	24	17
Total	**169**	**143**	**150**	**227**	**218**	**183**

Experiments



Thank your for your attention.

Do you have any questions?