Searching for Plans with Carefully Designed Probes

Nir Lipovetzky[1] Héctor Geffner[2]

DTIC
Universitat Pompeu Fabra[1]

ICREA & DTIC
Universitat Pompeu Fabra[2]

June 15, 2011

Dual Search architectures

State-of-the-art planners such as FF, FD, and LAMA are made of two or more parts:

- One that is fast but **incomplete**, good for "easy problems" (helpful actions, EHC, etc.)
- The other that is slower but **complete** (Greedy Best First Search)

In this work we explore a different **dual-search architecture**

The Planner PROBE

- **PROBE** extends a standard **GBFS** with probes
- A **probe** is thrown from each expanded state *s* in GBFS
- **Probes** are single action sequences constructed **greedily**

Challenge

- Design of probes so that "easy problems" are solved with almost **no search**

Motivation

- Understand inferences necessary to accomplish this behavior

Visualizations of Probes

(Loading state space)

Coverage

Domain	I	FF	LAMA	PROBE	1P
Blocks World	50	42	50	50	50
Cyber	30	4	25	24	13
Depots	22	22	20	22	14
Driver	20	16	20	20	15
Freecell	20	20	20	18	7
Parc-Printer	30	30	24	27	21
Pegsol	30	30	30	29	1
Pipesworld-No-Tan	50	35	44	45	19
Pipesworld-Tan	50	22	39	41	16
Scanalyzer	30	30	28	28	26
SokoBan	30	27	26	14	0
Storage	30	18	18	21	15
TPP	30	28	30	30	30
Transport	30	29	30	30	24
Trucks	30	11	16	9	0
Woods	30	17	30	30	30
...	...	...	...	...	...
Total	980	827	879	900	683
Percentage		84%	89%	92%	70%

Definition and Computation of Probes (Sketch)

A probe is an **action sequence** constructed greedily from a seed state s:

- choosing the next subgoal to achieve, if previous subgoal achieved (or none selected)
- iteratively choosing the best action for current subgoal
- posting the reasons (causal commitments) for choosing an action, and keeping them

Probes only allowed to visit **new states** (not yet expanded)

Probes **succeed** if they get to the goal; else fail, dumping expanded nodes into OPEN list

Visualization of Successful Probe

(Loading Successful Probe)

Visualization of Failed Probe

(Loading Failed Probe)

Outline of the rest of the talk

1. Explain the computation of probes more precisely:
   - how **subgoals** identified and partially ordered
   - how next **subgoal** selected and filtered (consistency)
   - how **action** chosen to achieve next subgoal
   - how **commitments** generated, respected, and consumed
2. Empirical Results

Subgoals and Partial Ordering: Landmark Graph

(Loading Landmarks)

Identification and ordering of landmarks, based on previous work, resembling FF goal agenda

Subgoal Selection

Next Subgoal Selection

1. **Computes** the set S of first unachieved landmarks that are **consistent** in state s
2. **Selects** the landmark $p \in S$ nearest according to "the heuristic" as the next subgoal in state s

- Landmark p **inconsistent** in s roughly if:
  - go for p first
  - it needs to be undone, to go for next subgoals

We want to mantain subgoals!

Subgoal Selection

Next Subgoal Selection

1. **Computes** the set S of first unachieved landmarks that are **consistent** in state s
2. **Selects** the landmark $p \in S$ nearest according to "the heuristic" as the next subgoal in state s

- Landmark p **inconsistent** in s roughly if:
  - go for p first
  - it needs to be undone, to go for next subgoals

We want to mantain subgoals! Why?

Subgoal Selection

Next Subgoal Selection

1. **Computes** the set S of first unachieved landmarks that are **consistent** in state s
2. **Selects** the landmark $p \in S$ nearest according to "the heuristic" as the next subgoal in state s

- Landmark p **inconsistent** in s roughly if:
  - go for p first
  - it needs to be undone, to go for next subgoals

We want to mantain subgoals! Why? To find dynamically a subgoal serialization

Subgoal Selection

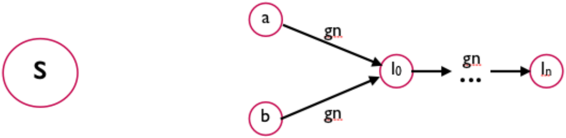
Next Subgoal Selection

1. **Computes** the set S of first unachieved landmarks that are **consistent** in state s
2. **Selects** the landmark $p \in S$ nearest according to "the heuristic" as the next subgoal in state s

- Landmark p **inconsistent** in s roughly if:
  - go for p first
  - it needs to be undone, to go for next subgoals

We want to mantain subgoals! Why? To find dynamically a subgoal serialization

Example of inconsistency



Question: achieve first *a* or *b* ?

If go for *a*:

- *a* is **inconsistent** if, once it is true, in order to achieve *b* , we need to undo *a*

Action Selection

Action Selection

1. **Computes** set of Helpful Actions in s for subgoal $\boldsymbol{g}$ excluding:
   - nodes already expanded
   - nodes from which top goal unreachable
2. **Selects** action $\boldsymbol{a}$ that gets **closer** to g according to the heuristic

Reasons for Action in Probes: Causal Commitments

Nodes in the Probes are actually pairs n = <s,C>, where a **Causal Commitment** $\langle \boldsymbol{a}, \boldsymbol{p}, \boldsymbol{B} \rangle$ is a constraint that states:

1. Fluent ***p*** is added by action ***a*** in order to achieve (at least) one fluent in ***B***.
2. ***p*** should remain true until an action ***a′*** adds a fluent in ***B***.

Commitment Violation

An action a' **violates** a commitment $\langle \boldsymbol{a}, \boldsymbol{p}, \boldsymbol{B} \rangle$ in a state s, if a' deletes ***p*** but does not add a fluent in ***B***.

Causal Commitments

Causal commitments in C generated and consumed by actions:

Commitments Generation

Action $\boldsymbol{a}$ in probe *generates* commitments $\langle \boldsymbol{a}, \boldsymbol{p}, \boldsymbol{B} \rangle$ in n, where:

- $\boldsymbol{p}$ is added by $\boldsymbol{a}$
- $\boldsymbol{B}$ is the set of fluents added by **actions** in the **relaxed plan** in n that have $\boldsymbol{p}$ as a precondition.

Commitments Consumption

Action $\boldsymbol{a'}$ *consumes* a commitment $\langle \boldsymbol{a}, \boldsymbol{p}, \boldsymbol{B} \rangle$ in n when:

- $\boldsymbol{a'}$ adds a fluent in $\boldsymbol{B}$.

Heuristic $h(G|n)$ over nodes n=<s,C> estimates cost of reaching G from s, while respecting the commitments in C

Summary

Probe is an **action sequence**

Choose:

- next subgoal
- sequence of best actions
- causal commitments

Techniques:

- landmarks, subgoaling, decomposition, consistency, commitments, helpful actions

PROBE extends a standard **GBFS** with probes

Experiments

We compare *PROBE* to *FF* and *LAMA* – 2008. Planners are evaluated using the following settings:

- Time out after 30 minutes.
- Memory out after 2 Gb.
- Domains from previous IPCs.
- Total of 980 instances.

Empirical Results

		FF			LAMA			PROBE				1P
Domain	I	S	T	Q	S	T	Q	S	T	Q	#P	S
Blocks World	50	42	0.22	39	50	0.69	86	50	0.21	40	1.0	50
Cyber	30	4	0.74	30	25	48.48	30	24	1.46	30	111.5	13
Depots	22	22	38.28	47	20	46.58	52	22	3.01	42	11.8	14
Freecell	20	20	2.81	55	20	19.78	64	18	45.45	67	35.1	7
Mystery	30	18	0.08	7	22	2.36	6	25	1.21	8	1.1	23
Parc-Printer	30	30	0.03	32	24	0.41	34	27	0.26	31	9.7	21
Pegsol	30	30	1.35	34	30	1.34	35	29	2.10	34	864.7	1
Pipes-No-Tan	50	35	0.45	28	44	1.04	37	45	0.35	33	6.4	19
Pipes-Tan	50	22	62.23	30	39	32.41	31	41	59.14	55	108.7	16
Scanalyzer	30	30	1.89	24	28	8.52	24	28	6.15	24	2.8	26
SokoBan	30	27	0.82	141	26	3.52	138	14	96.71	160	11,120.6	0
Storage	30	18	49.90	16	18	1.62	20	21	0.08	15	2.5	15
Transport	30	29	133.52	28	30	41.23	27	30	42.27	26	1.2	24
Trucks	30	11	5.66	23	16	0.61	24	9	20.55	26	2,818.4	0
Woods	30	17	0.26	117	30	5.84	100	30	5.45	154	1.0	30
...	...	...	...	...	...	...	...	...	...	...	...	...
Total	980	827	14.77	54	879	8.75	56	900	15.26	61		683
Percentage		84%			89%			92%				70%

Evaluating Different Techniques used in Probe

Feature Off	S	1P	Q	T
None	**92%**	**70%**	**67.0**	**34.8**
Probes	75%	–	71.0	99.6
Consistency	91%	40%	91.4	56.9
Subgoaling	86%	44%	80.7	55.2
Commitments	90%	63%	85.0	39.0

Table: Turning off different features in PROBE

- **Probes** helps significantly along all relevant dimensions.
- **Subgoaling** helps only when used in combination with the **consistency** tests

Evaluating Different Techniques used in Probe

Feature Off	S	1P	Q	T
None	**92%**	**70%**	**67.0**	**34.8**
Probes	75%	–	71.0	99.6
Consistency	91%	40%	91.4	56.9
Subgoaling	86%	44%	80.7	55.2
Commitments	90%	63%	85.0	39.0

Table: Turning off different features in PROBE

- **Probes** helps significantly along all relevant dimensions.
- **Subgoaling** helps only when used in combination with the **consistency** tests
- **Commitments** help mainly to improve the quality of the plans

Evaluating Different Techniques used in Probe

Feature Off	S	1P	Q	T
None	**92%**	**70%**	**67.0**	**34.8**
Probes	75%	–	71.0	99.6
Consistency	91%	40%	91.4	56.9
Subgoaling	86%	44%	80.7	55.2
Commitments	90%	63%	85.0	39.0

Table: Turning off different features in PROBE

- **Probes** helps significantly along all relevant dimensions.
- **Subgoaling** helps only when used in combination with the **consistency** tests
- **Commitments** help mainly to improve the quality of the plans

Summary

We have formulated:

- Planner PROBE extends standard GBFS with probes
- **Probes** provide fast, focused and effective look ahead
- Techniques: landmarks, subgoaling, decomposition, consistency, commitments, helpful actions
- PROBE competitive with state or the art planners
- A single probe solves 70% of IPC problems

Success of probes suggests many domains can be **solved easily** once a suitable **serialization** is found

Future Work

- **Which** methods are best for **finding/exploiting** good serializations

It may be worth to **revisit early work** in problem **decomposition** and goal **serialization** (e.g., Korf 87) in light of recent advances in planning (heuristics, landmarks, etc)

The **challenge** is to automatically recognize and exploit the structure of problems that are **nearly-decomposable**, even if not perfectly-decomposable (e.g., 15-puzzle).

Questions?

Thank you!

The heuristic $h(G|s, C)$ takes the commitments into account and is defined like the standard h_{add} where:

$$h(a|s, C) = \delta(a, s, C) + h(Pre(a)|s, C) \ . \quad (1)$$

$\delta(a, s, C)$

The **offset** of an action a is the **cost of achieving** the most "expensive" **violated** commitment $\langle a_i, p_i, B_i \rangle$ in C

As a result of the offset:

- **Applicable actions** a that violate a commitment may get $\boldsymbol{h(a|s, C) > 0}$.
- A **goal** G **reachable** in s may get $\boldsymbol{h(G|s, C) = \infty}$.