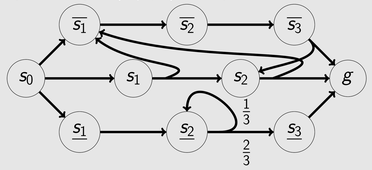
Probabilistic Replanning with Guarantees

Johannes Schmalz and Felipe Trevizan

Australian National University

Stochastic Shortest Path Problems (SSPs)

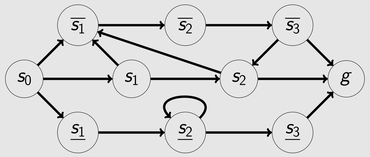
SSPs are a generalisation of classical planning, where each action may have a probabilistic effect – transitions are replaced by $P(s'|s,a)$, the probability of reaching s' after applying a.



This SSP will be a running example. Unless otherwise stated, its actions have unit cost and prob. effects have equal probabilities of reaching any effect.

Determinisation

We use the all-outcomes determinisation, where each probabilistic action a gets split into action $a \triangleright s'$ for each state s' such that $P(s'|s,a) > 0$.



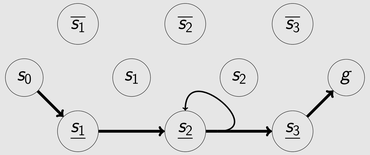
Now, we have a classical planning problem.

Replanners

We consider probabilistic replanners to be domain-independent planners that are designed to output policies in an online manner. The state-of-the-art replanner is **Robust-FF** (winner of IPPC 2008), which uses plans from the all-outcomes determinisation to form a policy.

Policies and Network Flow

A policy induces a solution to the network flow problem across an SSP, and vice versa.

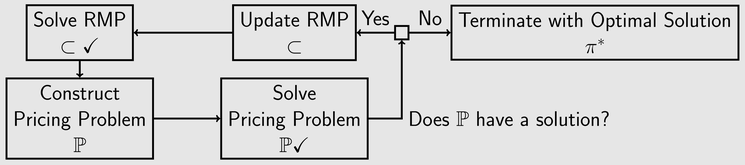


$$\cong \qquad \pi(s) = \begin{cases} \underline{a_0} & \text{if } s = s_0 \\ \underline{a_i} & \text{if } s = \underline{s_i} \end{cases}$$

We can find the optimal policy by finding the minimal combination of plans and cycles that form a policy i.e. a flow on the SSP; but doing this naively is infeasible, we would need a variable for each plan and cycle!

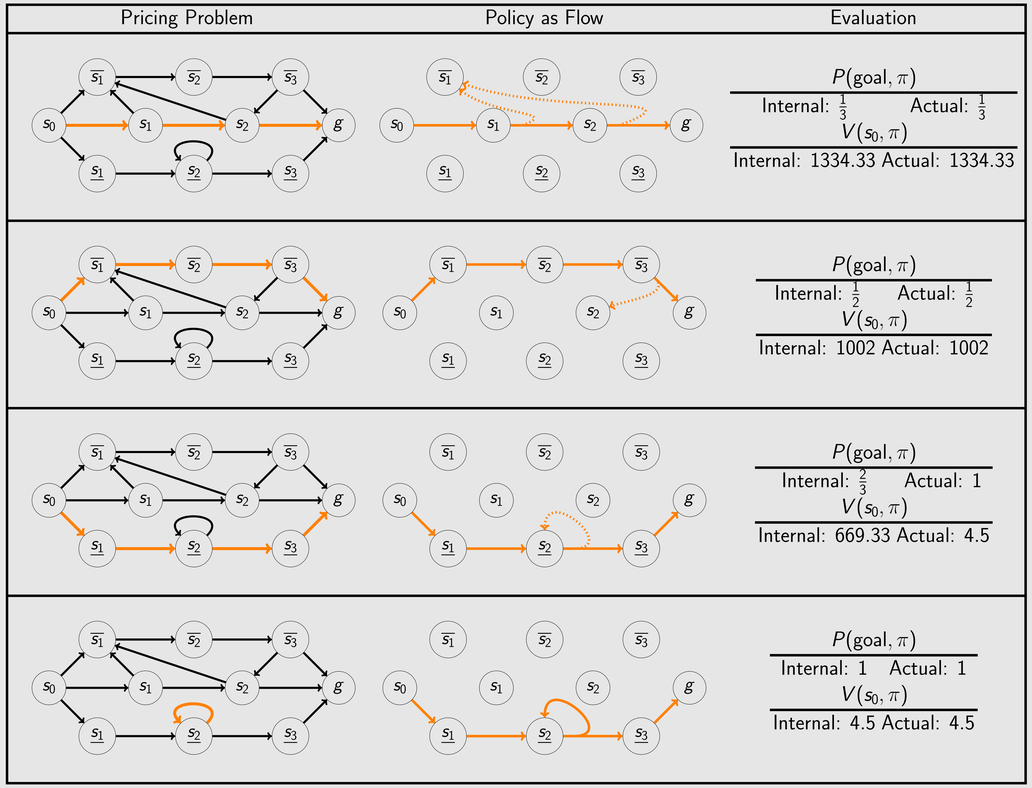
Column Generation and CoGNeRe

Given a very large LP, column generation considers a similar LP but with only a subset of variables, called the **Reduced Master Problem (RMP)**, which for us contains some plans and cycles. Column Generation lets us iteratively add plans and cycles that will bring the RMP closer to the original problem's optimal objective as follows:

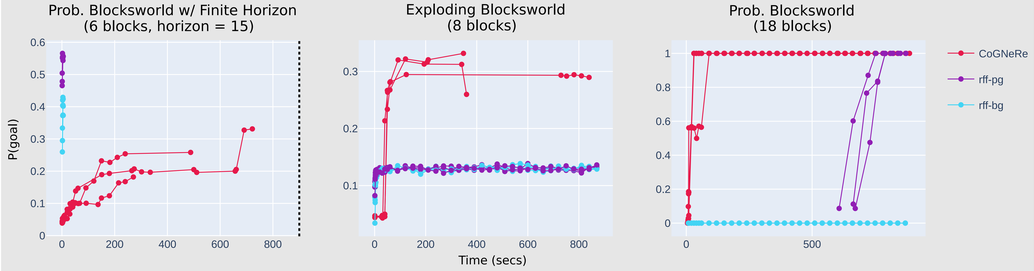


♠ Column generation over network flow LP gives CoGNeRe.

CoGNeRe in Action (Column Generation over Network Flow for Replanning)



Results (Selective and Preliminary)



Graphs show $P(\text{goal}|\pi)$ where π is replanner's online policy at given time (secs). We consider CoGNeRe and Robust-FF on some IPPC domains.

Pricing Problem

For us, the pricing problem is the all-outcomes determinisation but with different action costs; and on this new problem we must find a negative cost plan (not necessarily the cheapest), or a negative cycle. The following features make it non-trivial to adapt heuristics

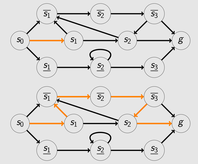
- state-dependent costs ($C(s,a) \neq C(s',a)$);
- negative costs (and the presence of negative cycles).

Bellman-Ford or Dijkstra's algorithm with some modifications work, but they do not scale up.

Large Neighbourhood Search

We are looking for some plan that is cheap enough, so we can use a variant of Large Neighbourhood Search:

Select a plan as the incumbent solution.



"Destroy and Repair" the plan; we do this by selecting a state in the plan as an artificial starting state, marking successors in the plan as artificial goals, and then running a depth-bounded search. If a cheaper plan is found, set that as the new incumbent solution.

Envelope Planner

Negative costs can only appear on transitions (s,a,s') where s,a appears in a plan or cycle in the RMP. So, we can run Bellman-Ford on relevant states only, and then run A* from the fringe to goals, using classical heuristics for the all-outcomes determinisation.

Closing the RMP—Policy-Objective Gap

In the example we see that CoGNeRe may have a closed policy ($P(\text{goal}|\pi) = 1$) without being aware of it. To mitigate this, when detected:

We start with an RMP with its reduced set of plans and cycles.

Extract the current best policy from the RMP and evaluate it as flow.

Now, decompose the flow into plans and cycles.

Bonus Facts

- CoGNeRe can be initialised with a set of plans and cycles or a policy – so you can kickstart it with Robust-FF or any other non-optimal planner.
- Thanks to column generation, we can give lower bounds for the optimal policy cost.

More information available at `schmlz.github.io/cognere`