

Planning and Optimization

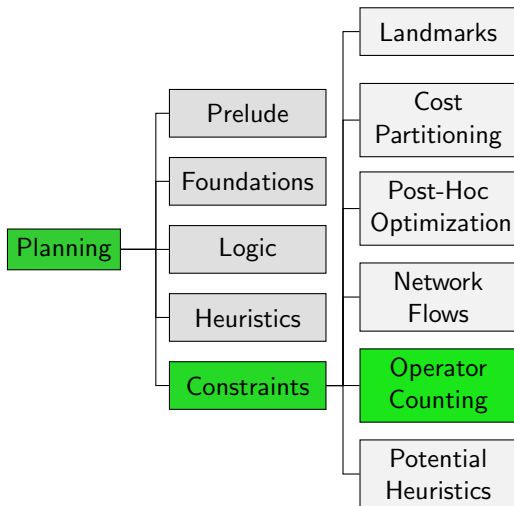
G11. Operator Counting

Malte Helmert and Gabriele Röger

Universität Basel

December 14, 2022

Content of this Course



Introduction

Reminder: Flow Heuristic

In the previous chapter, we used **flow constraints** to describe **how often operators must be used** in each plan.

Example (Flow Constraints)

Let Π be a planning problem with operators $\{O_{\text{red}}, O_{\text{green}}, O_{\text{blue}}\}$.
The flow constraint for some atom a is the constraint

$$1 + \text{Count}_{O_{\text{green}}} = \text{Count}_{O_{\text{red}}} \text{ if}$$

- a is true in the initial state
- a is false in the goal
- O_{green} produces a
- O_{red} consumes a

In natural language, the flow constraint expresses that

Reminder: Flow Heuristic

In the previous chapter, we used **flow constraints** to describe **how often operators must be used** in each plan.

Example (Flow Constraints)

Let Π be a planning problem with operators $\{O_{\text{red}}, O_{\text{green}}, O_{\text{blue}}\}$.
The flow constraint for some atom a is the constraint

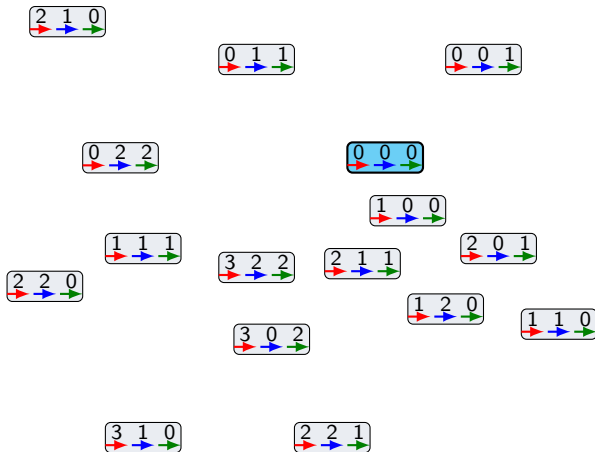
$$1 + \text{Count}_{O_{\text{green}}} = \text{Count}_{O_{\text{red}}} \text{ if}$$

- a is true in the initial state
- O_{green} produces a
- a is false in the goal
- O_{red} consumes a

In natural language, the flow constraint expresses that
every plan uses O_{red} **once more** than O_{green} .

Reminder: Flow Heuristic

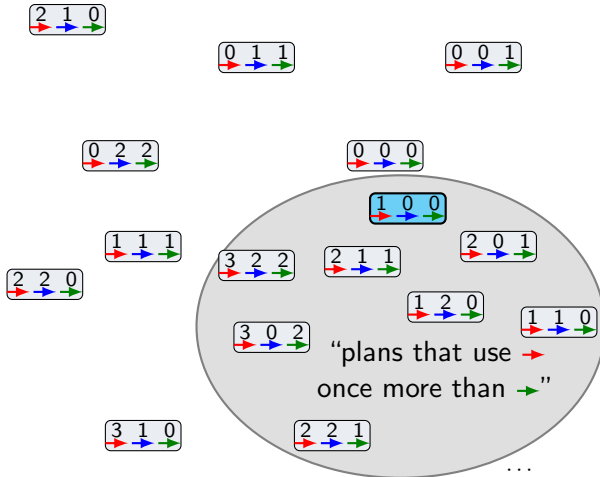
Let us now observe how each flow constraint alters the operator count solution space.



...

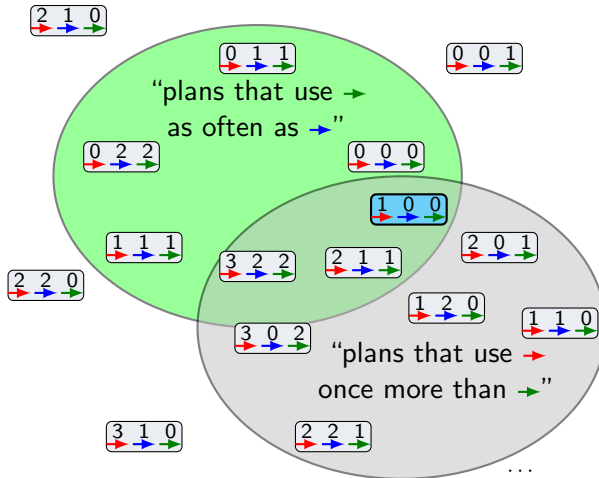
Reminder: Flow Heuristic

Let us now observe how each flow constraint alters the operator count solution space.



Reminder: Flow Heuristic

Let us now observe how each flow constraint alters the operator count solution space.



Operator-counting Framework

Operator Counting

Operator counting

- generalizes this idea to a framework that allows to **admissibly combine different heuristics**.
- uses **linear constraints** ...
- ... that describe **number of occurrences** of an operator ...
- ... and must be satisfied by **every plan**.
- provides declarative way to describe **knowledge about solutions**.
- allows **reasoning about solutions** to derive heuristic estimates.

Operator-counting Constraint

Definition (Operator-counting Constraints)

Let Π be a planning task with operators O and let s be a state. Let \mathcal{V} be the set of integer variables Count_o for each $o \in O$.

A linear inequality over \mathcal{V} is called an **operator-counting constraint** for s if for every plan π for s setting each Count_o to the number of occurrences of o in π is a feasible variable assignment.

Operator-counting Heuristics

Definition (Operator-counting IP/LP Heuristic)

The operator-counting integer program IP_C for a set C of operator-counting constraints for state s is

$$\text{Minimize } \sum_{o \in O} \text{cost}(o) \cdot \text{Count}_o \quad \text{subject to}$$

$$C \text{ and } \text{Count}_o \geq 0 \text{ for all } o \in O,$$

where O is the set of operators.

The **IP heuristic** h_C^{IP} is the objective value of IP_C ,
the **LP heuristic** h_C^{LP} is the objective value of its LP-relaxation.
If the IP/LP is infeasible, the heuristic estimate is ∞ .

Operator-counting Constraints

- Adding more constraints can only remove feasible solutions.
 - Fewer feasible solutions can only increase the objective value.
 - Higher objective value means better informed heuristic
- ⇒ Have we already seen other operator-counting constraints?

Reminder: Minimum Hitting Set for Landmarks

Variables

Non-negative variable Applied_o for each operator o

Objective

Minimize $\sum_o \text{cost}(o) \cdot \text{Applied}_o$

Subject to

$$\sum_{o \in L} \text{Applied}_o \geq 1 \text{ for all landmarks } L$$

Operator Counting with Disjunctive Action Landmarks

Variables

Non-negative variable Count_o for each operator o

Objective

Minimize $\sum_o \text{cost}(o) \cdot \text{Count}_o$

Subject to

$$\sum_{o \in L} \text{Count}_o \geq 1 \text{ for all landmarks } L$$

Reminder: Post-hoc Optimization Heuristic

For set of abstractions $\{\alpha_1, \dots, \alpha_n\}$:

Variables

Non-negative variables X_o for all operators $o \in O$

X_o is cost incurred by operator o

Objective

Minimize $\sum_{o \in O} X_o$

Subject to

$$\sum_{o \in O: o \text{ relev. for } \alpha} X_o \geq h^\alpha(s) \quad \text{for } \alpha \in \{\alpha_1, \dots, \alpha_n\}$$
$$X_o \geq 0 \quad \text{for all } o \in O$$

Operator Counting with Post-hoc Optimization Constraints

For set of abstractions $\{\alpha_1, \dots, \alpha_n\}$:

Variables

Non-negative variables Count_o for all operators $o \in O$

$\text{Count}_o \cdot \text{cost}(o)$ is cost incurred by operator o

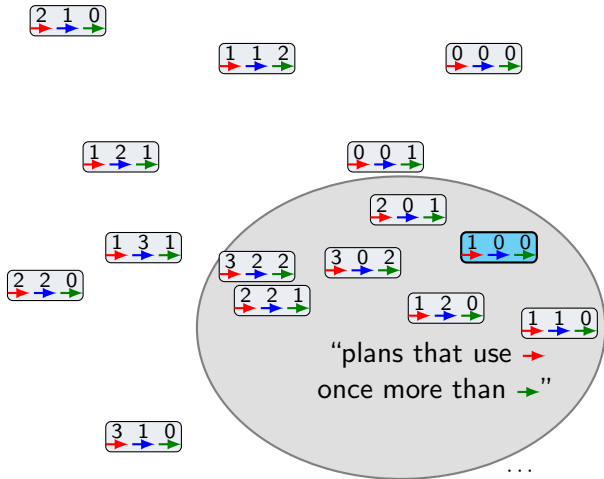
Objective

Minimize $\sum_{o \in O} \text{cost}(o) \cdot \text{Count}_o$

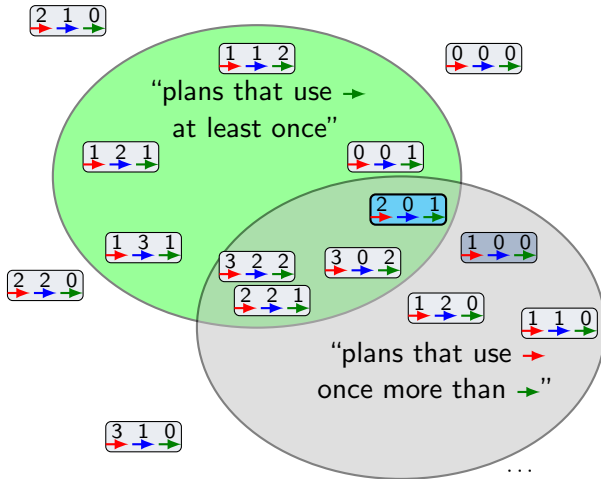
Subject to

$$\sum_{o \in O: o \text{ relev. for } \alpha} \text{cost}(o) \cdot \text{Count}_o \geq h^\alpha(s) \quad \text{for } \alpha \in \{\alpha_1, \dots, \alpha_n\}$$
$$\text{cost}(o) \cdot \text{Count}_o \geq 0 \quad \text{for all } o \in O$$

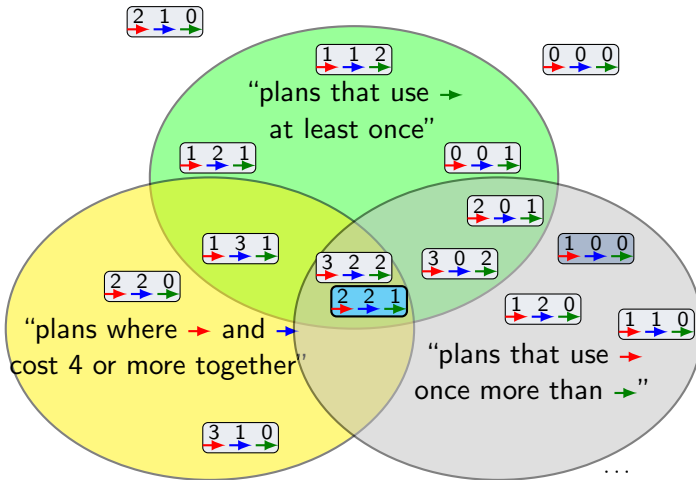
Example



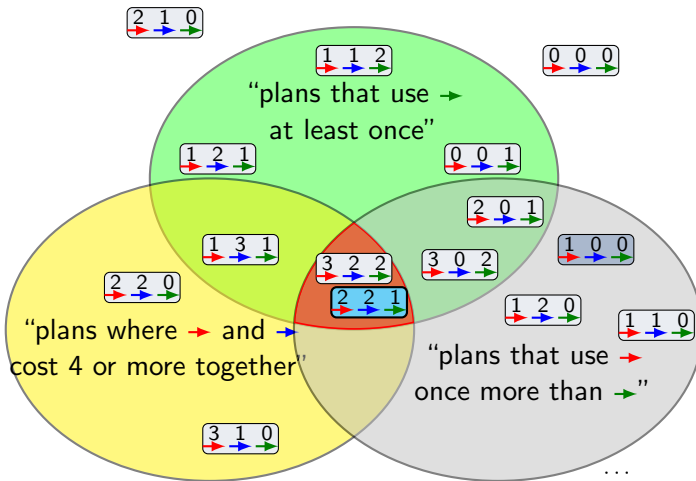
Example



Example



Example



Further Examples?

- The definition of operator-counting constraints can be extended to groups of constraints and auxiliary variables.
- With this extended definition we could also cover more heuristics, e.g., the perfect relaxation heuristic h^+

Properties

Admissibility

Theorem (Operator-counting Heuristics are Admissible)

*The IP and the LP heuristic are **admissible**.*

Proof.

Let C be a set of operator-counting constraints for state s and π be an optimal plan for s . The number of operator occurrences of π are a feasible solution for C . As the IP/LP minimizes the total plan cost, the objective value cannot exceed the cost of π and is therefore an admissible estimate. □

Dominance

Theorem

Let C and C' be sets of operator-counting constraints for s and let $C \subseteq C'$. Then $IP_C \leq IP_{C'}$ and $LP_C \leq LP_{C'}$.

Proof.

Every feasible solution of C' is also feasible for C . As the LP/IP is a minimization problem, the objective value subject to C can therefore not be larger than the one subject to C' . □

Adding more constraints can only improve the heuristic estimate.

Heuristic Combination

Operator counting as heuristic combination

- Multiple operator-counting heuristics can be combined by computing h_c^{LP} / h_c^{IP} for the union of their constraints.
- This is an admissible combination.
 - Never worse than maximum of individual heuristics
 - Sometimes even better than their sum
- We already know a way of admissibly combining heuristics: cost partitioning.
⇒ How are they related?

Connection to Cost Partitioning

Theorem

Let C_1, \dots, C_n be sets of operator-counting constraints for s and $\mathcal{C} = \bigcup_{i=1}^n C_i$. Then $h_{\mathcal{C}}^{\text{LP}}$ is the *optimal general cost partitioning* over the heuristics $h_{C_i}^{\text{LP}}$.

Proof Sketch.

In $\text{LP}_{\mathcal{C}}$, add variables Count_o^i and constraints $\text{Count}_o = \text{Count}_o^i$ for all operators o and $1 \leq i \leq n$. Then replace Count_o by Count_o^i in C_i .

Dualizing the resulting LP shows that $h_{\mathcal{C}}^{\text{LP}}$ computes a cost partitioning. Dualizing the component heuristics of that cost partitioning shows that they are $h_{C_i}^{\text{LP}}$.

Comparison to Optimal Cost Partitioning

- some heuristics are **more compact** if expressed as operator counting
- some heuristics **cannot be expressed** as operator counting
- **operator counting IP** even better than optimal cost partitioning
- Cost partitioning maximizes, so heuristics must be encoded perfectly to guarantee admissibility.
Operator counting minimizes, so missing information just makes the heuristic weaker.

Summary

Summary

- Many heuristics can be formulated in terms of **operator-counting constraints**.
- The operator counting heuristic framework allows to **combine the constraints** and to reason on the entire encoded declarative knowledge.
- The heuristic estimate for the combined constraints **can be better than the one of the best ingredient heuristic** but never worse.
- Operator counting is **equivalent to optimal general cost partitioning** over individual constraints.