# Planning and Optimization

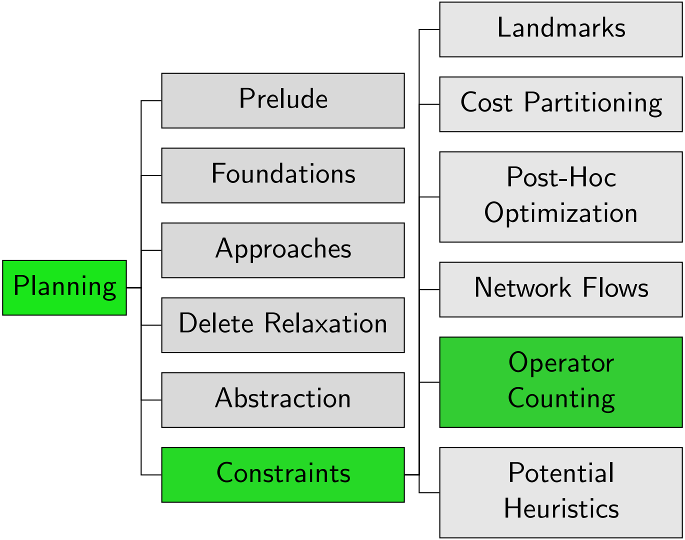
## F11. Operator Counting

Malte Helmert and Gabriele Röger

Universität Basel

December 16, 2024

# Content of the 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

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

- *a* is true in the initial state
- *a* is false in the goal
- $o_{\text{green}}$ produces *a*
- $o_{\text{red}}$ 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

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

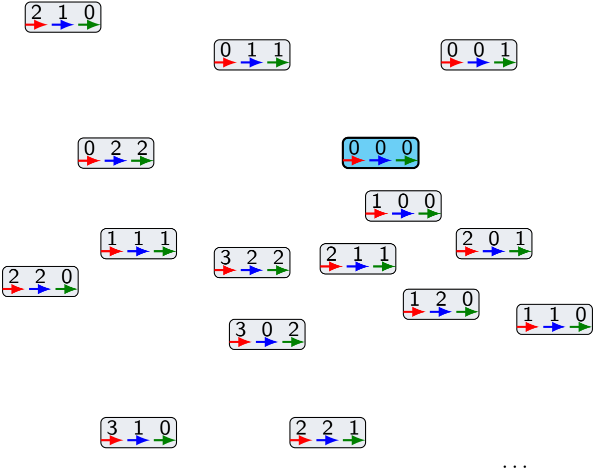
- $a$ is true in the initial state
- $a$ is false in the goal
- $o_{\text{green}}$ produces $a$
- $o_{\text{red}}$ consumes $a$

In natural language, the flow constraint expresses that

every plan uses $o_{\text{red}}$ once more than $o_{\text{green}}$.

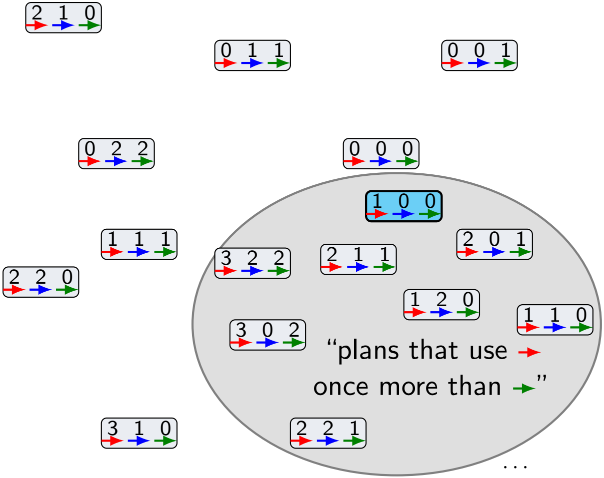
# Reminder: Flow Heuristic

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



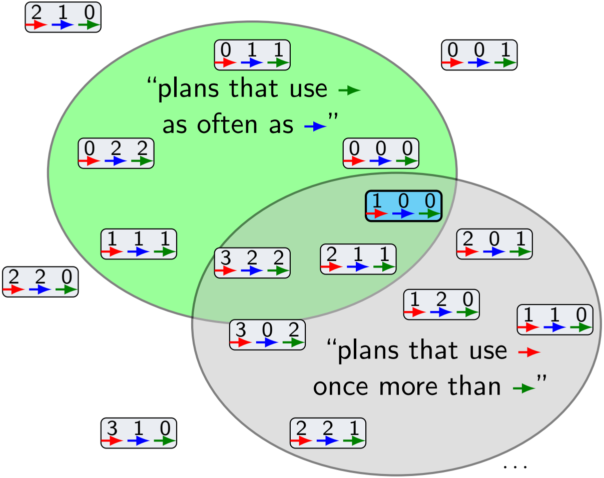
# Reminder: Flow Heuristic

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

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

# Operator-counting Heuristics

**Definition (Operator-counting IP/LP Heuristic)**

The operator-counting integer program $\mathrm{IP}_C$ for a set $C$ of operator-counting constraints for state $s$ is

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

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

where $O$ is the set of operators.

The IP heuristic $h^{\mathrm{IP}}_C$ is the objective value of $\mathrm{IP}_C$,
the LP heuristic $h^{\mathrm{LP}}_C$ is the objective value of its LP-relaxation.

If the IP/LP is infeasible, the heuristic estimate is $\infty$.

# 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 $\text{Applied}_o$ for each operator $o$

**Objective**

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

**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 $\text{Count}_o$ for each operator $o$

## Objective

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

## 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**

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

# Operator Counting with Post-hoc Optimization Constraints

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

## Variables

Non-negative variables $\text{Count}_o$ for all operators $o \in O$
$\text{Count}_o \cdot \mathit{cost}(o)$ is cost incurred by operator $o$

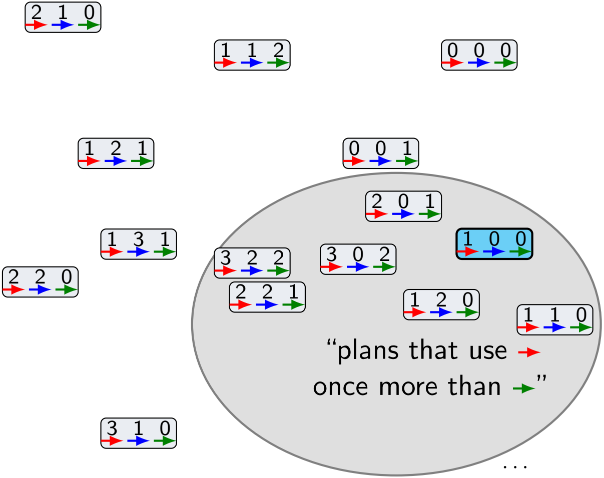
## Objective

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

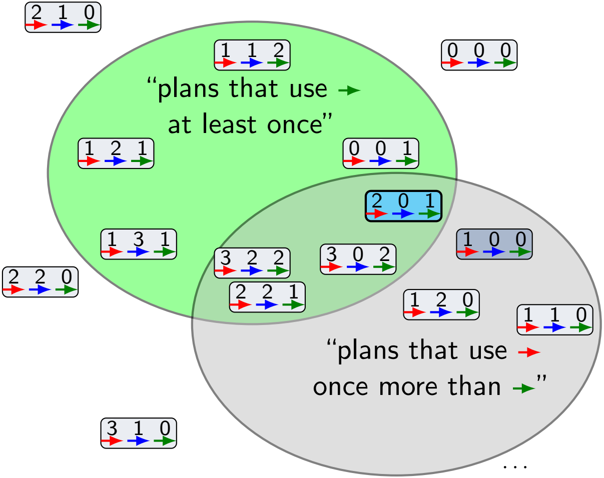
## Subject to

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

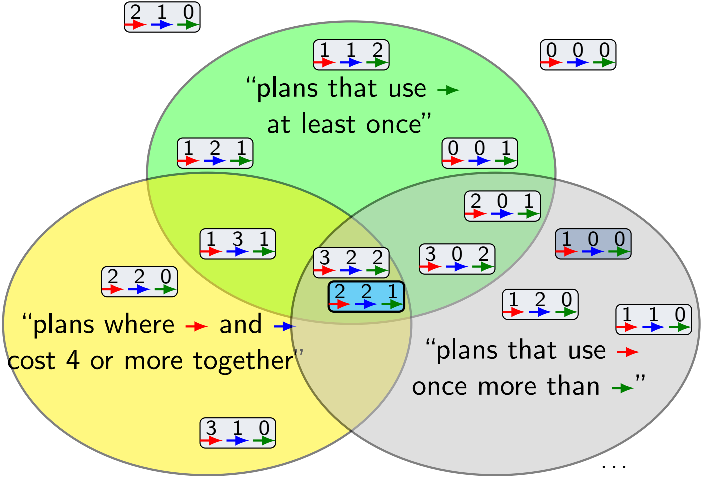
# Example


2 1 0
1 1 2
0 0 0
1 2 1
0 0 1
2 0 1
1 3 1
1 0 0
3 2 2
3 0 2
2 2 0
2 2 1
1 2 0
1 1 0
“plans that use → once more than →”
3 1 0
...

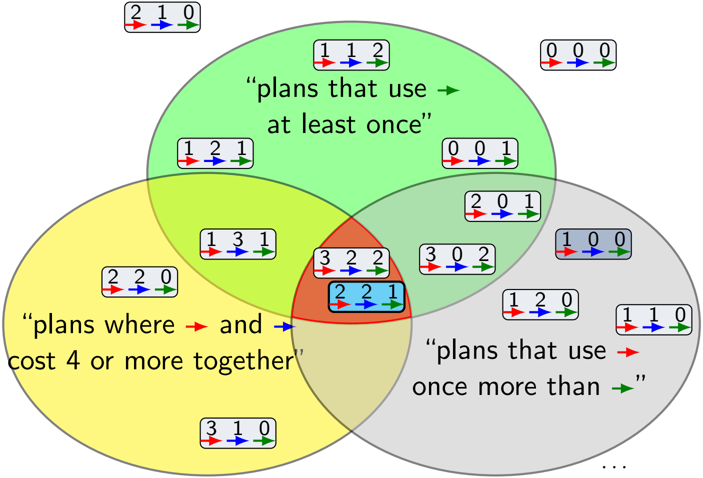
# Example


2 1 0
1 1 2
0 0 0
"plans that use → at least once"
1 2 1
0 0 1
2 0 1
1 3 1
1 0 0
3 2 2
3 0 2
2 2 0
2 2 1
1 2 0
1 1 0
"plans that use → once more than →"
3 1 0
...

# Example


2 1 0
1 1 2
0 0 0
"plans that use → at least once"
1 2 1
0 0 1
2 0 1
1 3 1
3 2 2
3 0 2
1 0 0
2 2 0
2 2 1
1 2 0
1 1 0
"plans where → and → cost 4 or more together"
"plans that use → once more than →"
3 1 0
. . .

# Example



“plans that use → at least once”

“plans where → and → cost 4 or more together”

“plans that use → once more than →”

# 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 $\pi$ be an optimal plan for $s$. The number of operator occurrences of $\pi$ are a feasible solution for $C$. As the IP/LP minimizes the total plan cost, the objective value cannot exceed the cost of $\pi$ 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* $\mathrm{IP}_C \leq \mathrm{IP}_{C'}$ *and* $\mathrm{LP}_C \leq \mathrm{LP}_{C'}$.

**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 $h^{\mathrm{LP}}_{\mathcal{C}}/h^{\mathrm{IP}}_{\mathcal{C}}$ 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.

  $\Rightarrow$ 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 $h^{\mathrm{LP}}_{\mathcal{C}}$ is the optimal general cost partitioning over the heuristics $h^{\mathrm{LP}}_{C_i}$.*

## Proof Sketch.

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

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

# 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.