Planning and Optimization

D4. Operator Counting

Malte Helmert and Gabriele Röger

Universität Basel

December 8, 2016

Operator-counting Framework

Reminder: Optimal Cost Partitioning for Landmarks

Variables

Occurrences_o for each operator o

Objective

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

Subject to

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

$$\text{Occurrences}_o \geq 0 \text{ for all operators } o$$

Numbers of operator occurrences in any plan satisfy constraints.
Minimizing the total plan cost gives an admissible estimate.
Can we apply this idea more generally?

Operator Counting

Operator-counting Constraints

- linear constraints whose variables denote number of occurrences of a given operator
- must be satisfied by every plan

Examples:

- $Y_{o_1} + Y_{o_2} \geq 1$ "must use o_1 or o_2 at least once"
- $Y_{o_1} - Y_{o_3} \leq 0$ "cannot use o_1 more often than o_3"

Motivation:

- declarative way to represent knowledge about solutions
- allows reasoning about solutions to derive heuristic estimates

Operator Counting Heuristics

Operator occurrences in potential plans

(2,1,0)

(1,1,2) (0,0,0)

(0,0,1)

(1,2,1)

(3,0,2)

(1,3,1)

(3,2,2)

(2,2,0)

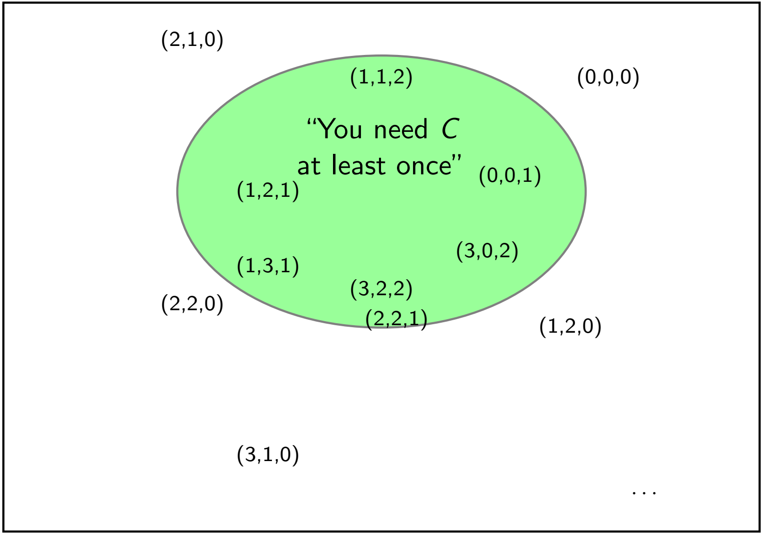
(2,2,1) (1,2,0)

(3,1,0)

...

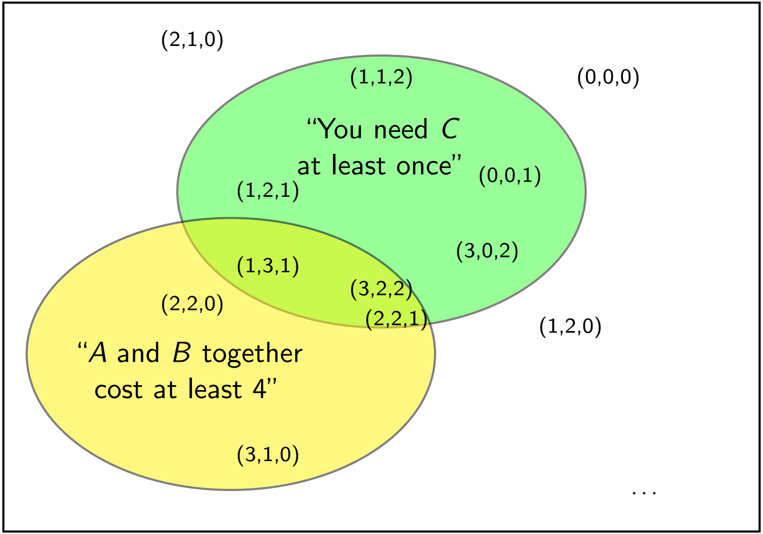
Operator Counting Heuristics

Operator occurrences in potential plans



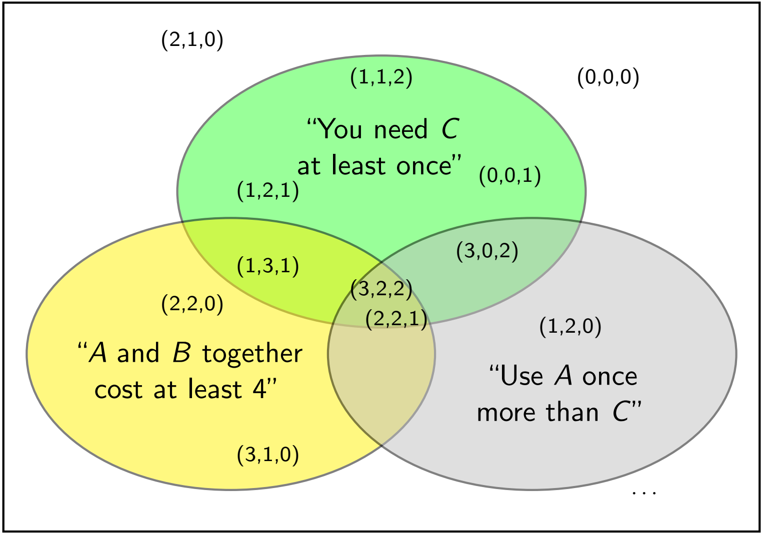
Operator Counting Heuristics

Operator occurrences in potential plans



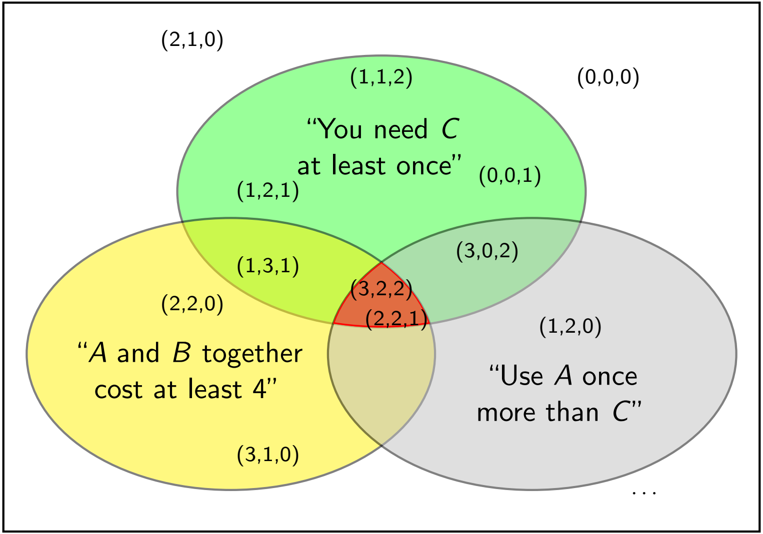
Operator Counting Heuristics

Operator occurrences in potential plans



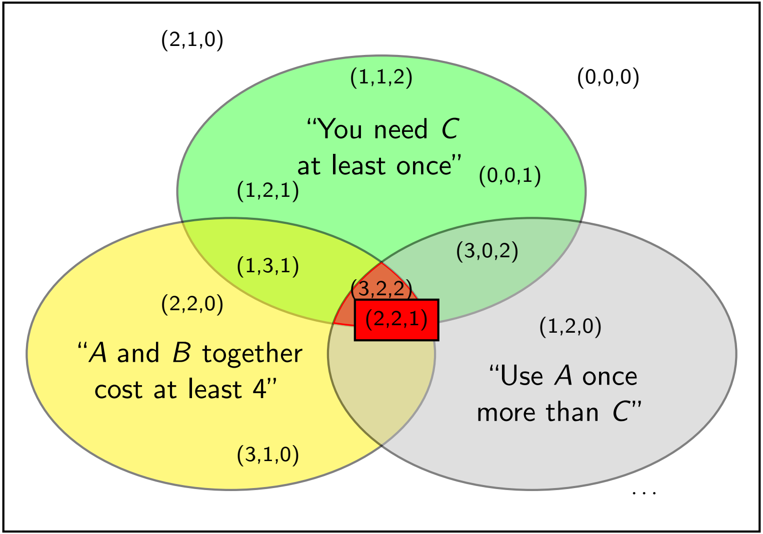
Operator Counting Heuristics

Operator occurrences in potential plans



Operator Counting Heuristics

Operator occurrences in potential plans



Operator-counting Constraint

Definition (Operator-counting Constraints)

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

A linear inequality over $\mathcal{Y}$ is called an operator-counting constraint for s if for every plan π for s setting each Y_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

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

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

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 LP/IP is infeasible, the heuristic estimate is ∞.

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

Proof.

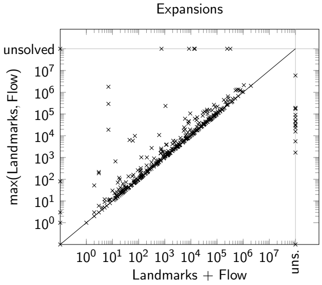
Every feasible solution of C' is also feasible for C. As the LP/IP is a minimization problem, the objective value subject to C can therefore not be larger than the one subject to C'. □

Adding more constraints can only improve the heuristic estimate.

Combining Heuristics

Combination of two heuristics

- Use both operator-counting constraints
- Combination always dominates individual heuristics
- Positive interaction between constraints



Combination often better than best individual heuristic

Constraints from Disjunctive Action Landmarks

Optimal cost partitioning for disjunctive action landmarks

- Use one landmark constraint per landmark

Landmark constraint for landmark L

$$\sum_{o \in L} Y_o \geq 1$$

Constraints from Flow Heuristic

Flow heuristic

- Use one flow constraint per atom

Flow Constraint for atom a

$$[a \in s] + \sum_{o \in O : a \in eff(o)} Y_o = [a \in \gamma] + \sum_{o \in O : a \in pre(o)} Y_o$$

Remark: Assumes transition normal form (not a limitation)

Constraints from Post-hoc Optimization Heuristic

Post-hoc optimization heuristic

- In chapter D3: X_o for cost incurred by operator o
- Replace each such variable with $Y_o \cdot cost(o)$ to fit the operator-counting framework.
- Use one post-hoc optimization constraint per sub-heuristic

Post-hoc optimization constraint for heuristic h

$$\sum_{o \text{ is relevant for } h} Y_o \cdot cost(o) \geq h(s)$$

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
  - optimal cost partitioning for abstractions, and
  - the perfect relaxation heuristic h^+.

Connection to Cost Partitioning

Operator-counting Heuristics and General Cost Partitioning

Theorem

Combining operator-counting heuristics in one LP
is equivalent to
computing their optimal general cost partitioning (gOCP).

Proof idea: The linear programs are each others duals.

Use the Theorem to Combine Heuristics

- Easy way to compute cost partitioning of heuristics
  - LP can be more compact (variable elimination)
  - No need for one variable per operator and subproblem
- Even better combination of heuristics with IP heuristic
  - Considers that operator cannot be used 1.5 times
  - But computation is no longer polynomial

Use the Theorem to Analyze Heuristics

Analyze operator counting heuristics

1. Group linear constraints into sets of operator-counting constraints
2. Figure out what heuristic is computed with just one such set
3. Your original operator-counting heuristic computes the optimal general cost partition of those component heuristics

Use the Theorem to Analyze Heuristics

Analyze operator counting heuristics

Example: flow heuristic

1. Group linear constraints into sets of operator-counting constraints
2. Figure out what heuristic is computed with just one such set
3. Your original operator-counting heuristic computes the optimal general cost partition of those component heuristics

Use the Theorem to Analyze Heuristics

Analyze operator counting heuristics

Example: flow heuristic

1. Group linear constraints into sets of operator-counting constraints
   - One group of flow constraints per variable
2. Figure out what heuristic is computed with just one such set
3. Your original operator-counting heuristic computes the optimal general cost partition of those component heuristics

Use the Theorem to Analyze Heuristics

Analyze operator counting heuristics

Example: flow heuristic

1. Group linear constraints into sets of operator-counting constraints
   - One group of flow constraints per variable
2. Figure out what heuristic is computed with just one such set
   - Minimizing total cost while respecting flow in projection to one variable
   - Shortest path in projection
3. Your original operator-counting heuristic computes the optimal general cost partition of those component heuristics

Use the Theorem to Analyze Heuristics

Analyze operator counting heuristics

Example: flow heuristic

1. Group linear constraints into sets of operator-counting constraints
   - One group of flow constraints per variable
2. Figure out what heuristic is computed with just one such set
   - Minimizing total cost while respecting flow in projection to one variable
   - Shortest path in projection
3. Your original operator-counting heuristic computes the optimal general cost partition of those component heuristics
   - Flow heuristic = gOCP(atomic projection heuristics)

Other Examples

What about the rest of our examples?

- Landmark constraints
  - gOCP(individual landmark heuristics)
- Post-hoc optimization heuristic
  - gOCP(heuristics that spend a minimum cost on relevant ops)
  - Also: cost partitioning over atomic projection heuristics
    - Operator costs not independent
    - Scale with one factor per projection

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.
- The combination into one operator-counting heuristic corresponds to the computation of the optimal general cost partitioning for the ingredient heuristics.

Literature (1)

References on the operator-counting framework:

Blai Bonet.
An Admissible Heuristic for SAS+ Planning Obtained from the State Equation.
Proc. IJCAI 2013, pp. 2268–2274, 2013.
Suggests combination of flow constraints and landmark constraints.

Tatsuya Imai and Alex Fukunaga.
A Practical, Integer-linear Programming Model for the Delete-relaxation in Cost-optimal Planning.
Proc. ECAI 2014, pp. 459–464, 2014.
IP formulation of h^+.

Literature (2)

Florian Pommerening, Gabriele Röger, Malte Helmert and Blai Bonet.
LP-based Heuristics for Cost-optimal Planning.
Proc. ICAPS 2014, pp. 226–234, 2014.
Systematic introduction of operator-counting framework.

Florian Pommerening, Malte Helmert, Gabriele Röger and Jendrik Seipp.
From Non-Negative to General Operator Cost Partitioning.
Proc. AAAI 2015, pp. 3335–3341, 2015.
Relation to general cost partitioning.