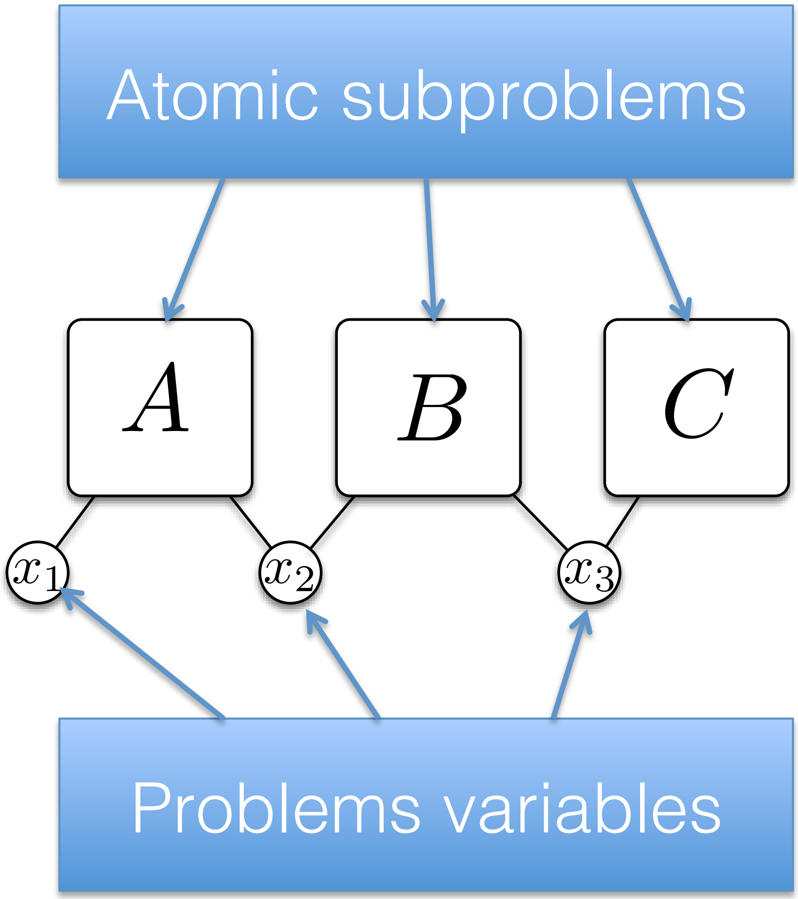
Dynamic Programming on Nominal Graphs

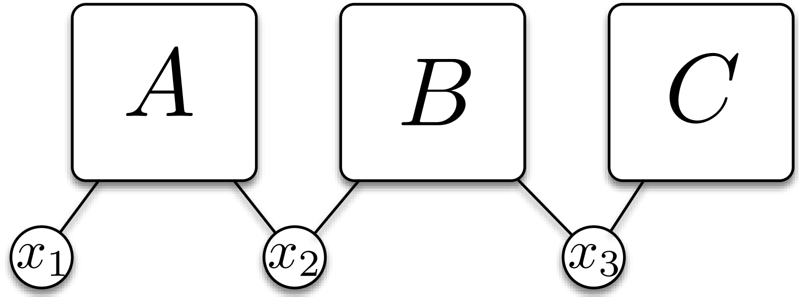
Matteo Sammartino
Radboud University

(joint work with Nicklas Hoch, Ugo Montanari)

Optimization problems as hypergraphs



Optimization problems as hypergraphs



- Lack of algebraic structure for resolution via structural recursion
- No indication of variable elimination order (secondary optimization problem)

Process calculus-like representation

$A(x_1, x_2)$

atomic problem *A* with two variables x_1, x_2

$A(x_1, x_2) \parallel B(x_2, x_3)$

composite problem

$(x_1)A(x_1, x_2)$

- Solved w.r.t. x_1 (elimination)
- Solution is parametric in x_2

Permutation algebras

Infinite set of variable names $\mathcal{N}$

Permutations (bijections) $\pi : \mathcal{N} \to \mathcal{N}$

Algebras P including permutations and axioms

$$p \,\mathsf{id} \equiv p \qquad (p\pi')\pi \equiv p(\pi \circ \pi')$$

Support

$A \subseteq \mathcal{N}$ supports $p \in P$ whenever

$$\forall \pi : \pi_{|A} = \mathrm{id} \implies p\,\pi = p$$

Minimal support $supp(p)$ generalizes free variables

An algebraic specification for optimization problems

$$p, q := p \parallel q \mid (x)p \mid p\pi \mid A(\tilde{x}) \mid nil$$

Axioms:

- usual process calculi ones
- permutations distribute over operations

For every term p:

$supp(p) = fn(p)$ (non-restricted variables)

Alfa-conversion

$$(x)p \equiv (y)p[x \mapsto y] \qquad y \notin supp(p)$$

Scope extension

$$(x)(p \parallel q) \equiv (x)p \parallel q \qquad x \notin supp(q)$$

Correct handling of restrictions in all algebras

Hierarchical specification

no scope extension

terms describe solutions of secondary optimization problem (variable elimination order)

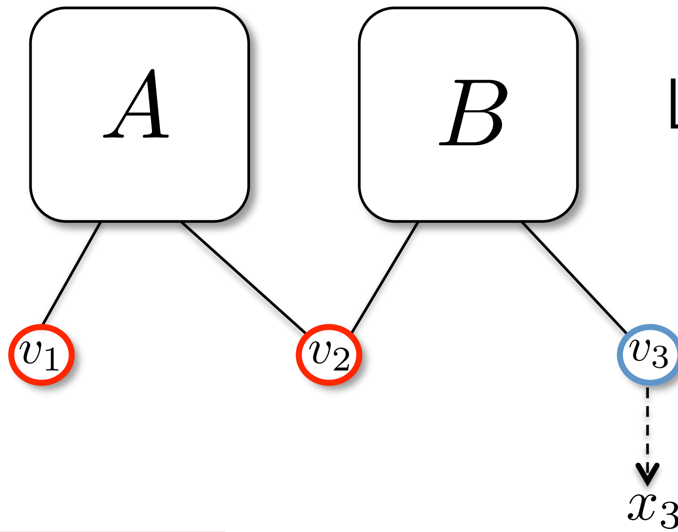
Normal form: elimination at the end

$$(x_1)(x_2)(x_3)(A(x_1,x_2) \parallel B(x_2,x_3) \parallel C(x_3))$$

Canonical form: elimination as soon as possible

$$(x_2)((x_1)A(x_1,x_2) \parallel (x_3)(B(x_2,x_3) \parallel C(x_3)))$$

Nominal Graphs



Labeled hypergraph

Partial function

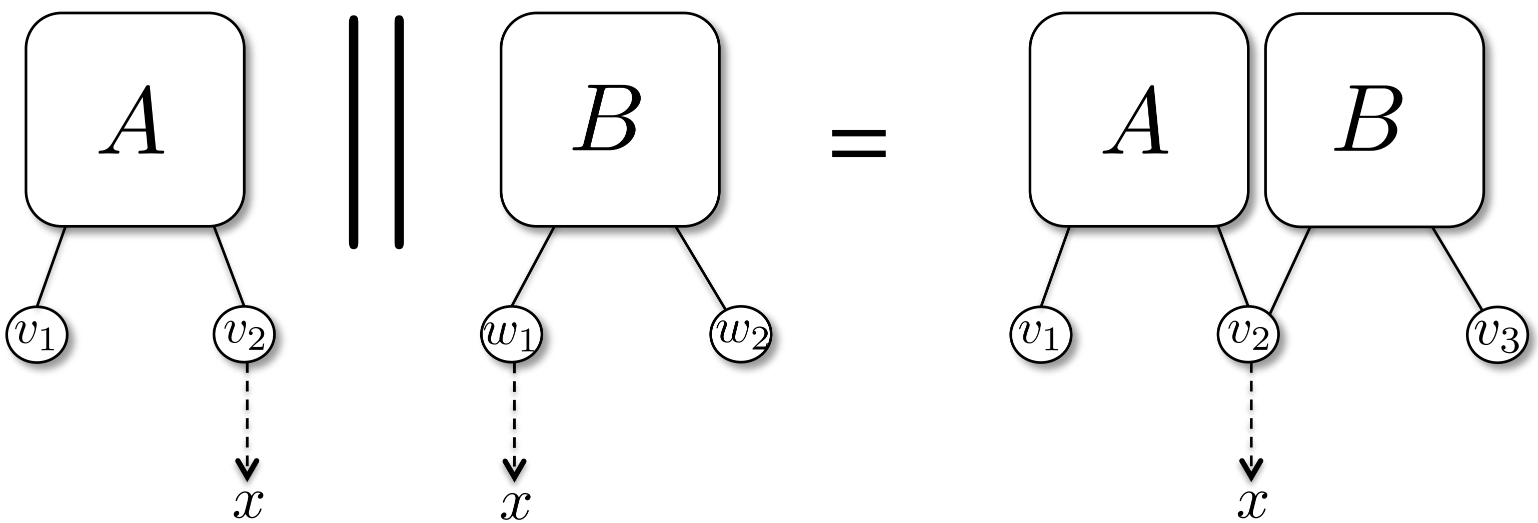
$$V \to \mathcal{N}$$

Non-interface vertices = restricted names (up to alfa-connversion)

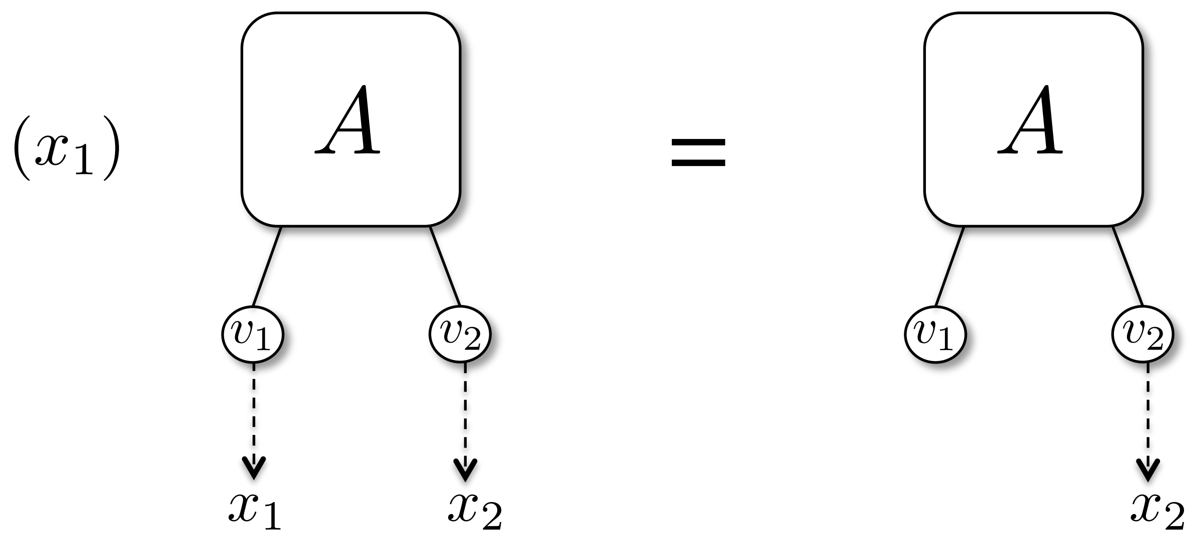
Interface vertices = free names

$$(x_1)(x_2)(A(x_1, x_2) \parallel B(x_2, x_3))$$

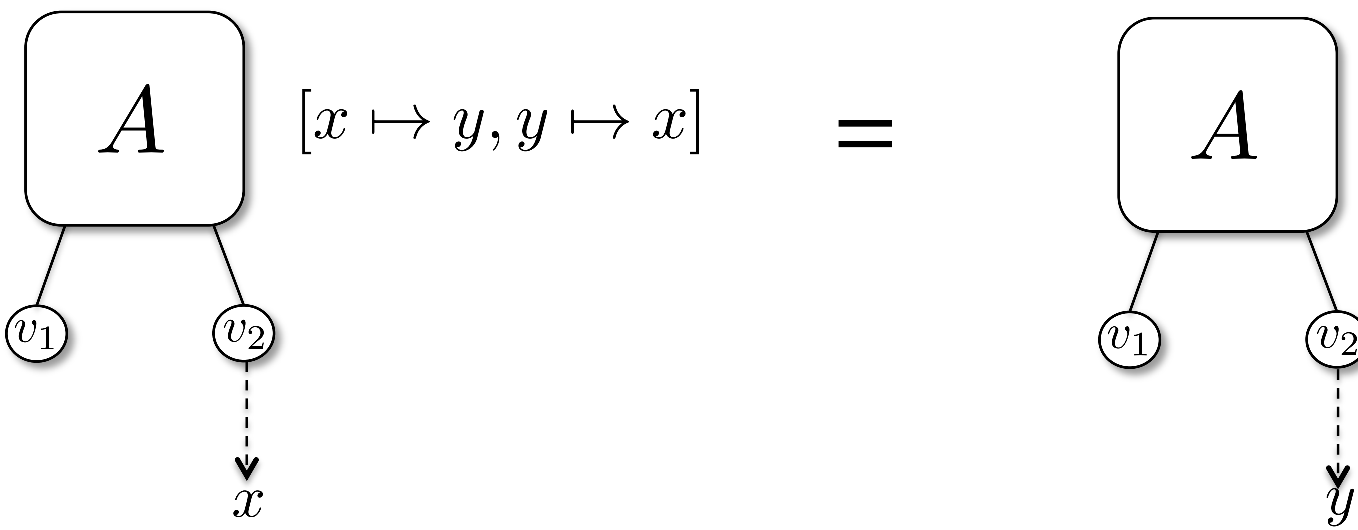
Optimization algebra of nominal graphs



Optimization algebra of nominal graphs



Optimization algebra of nominal graphs



supp = set of interface vertices

Optimization algebra of nominal graphs

Sound and complete axiomatization

congruent terms isomorphic nominal graphs

We have recursive evaluations of nominal graphs in any algebra (initial-algebra semantics)

Optimization algebra of cost functions

$$(\mathcal{N} \to \mathbb{D}) \to \mathbb{R}_\infty$$

Assignment of values to variables

cost

$[\![p]\!]^c$ = evaluation of p as a cost function

By general properties of permutation algebras:

$[\![p]\!]^c$ determined by assignments to $supp(p)$

Representable as a finite table

Computed via structural recursion
(“almost” dynamic programming)

Typical optimization problems

One cost function c_A for each atomic problem A

Goal: minimize total cost

$$[\![A\,(x_1,\ldots,x_n)]\!]^c\rho = c_A\,(\rho(x_1),\ldots,\rho(x_n))$$

$$[\![p_1 \parallel p_2]\!]^c\rho = [\![p_1]\!]^c\rho + [\![p_2]\!]^c\rho$$

$$[\![(x)p]\!]^c\rho = \min_{v\in\mathbb{D}} [\![p]\!]^c(\rho[x\mapsto v])$$

Typical optimization problems

One cost function c_A for each atomic problem A

Goal: minimize total cost

$[\![A$ … $x_n))$

When all variables are restricted

Cost function = minimal cost

$$[\![(x)p]\!]^c \rho = \min_{v \in \mathbb{D}} [\![p]\!]^c (\rho[x \mapsto v])$$

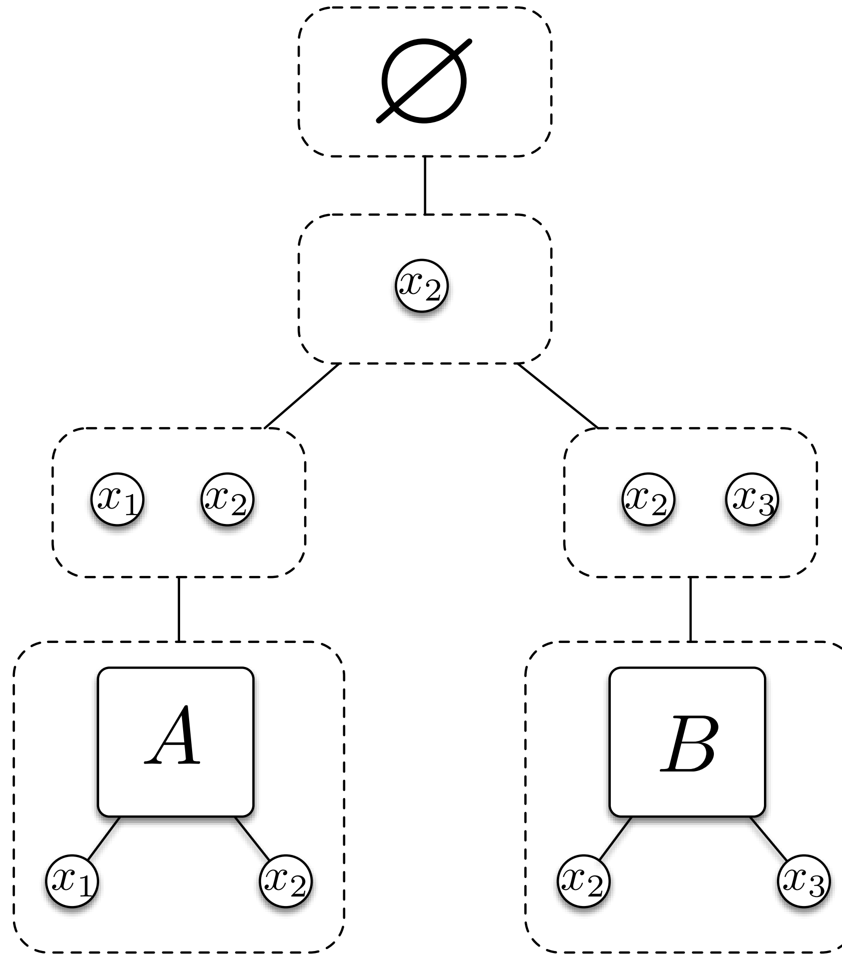
We can compute the cost function for a graph but…

How can we pick a variable elimination strategy?

Hierarchical nominal graphs

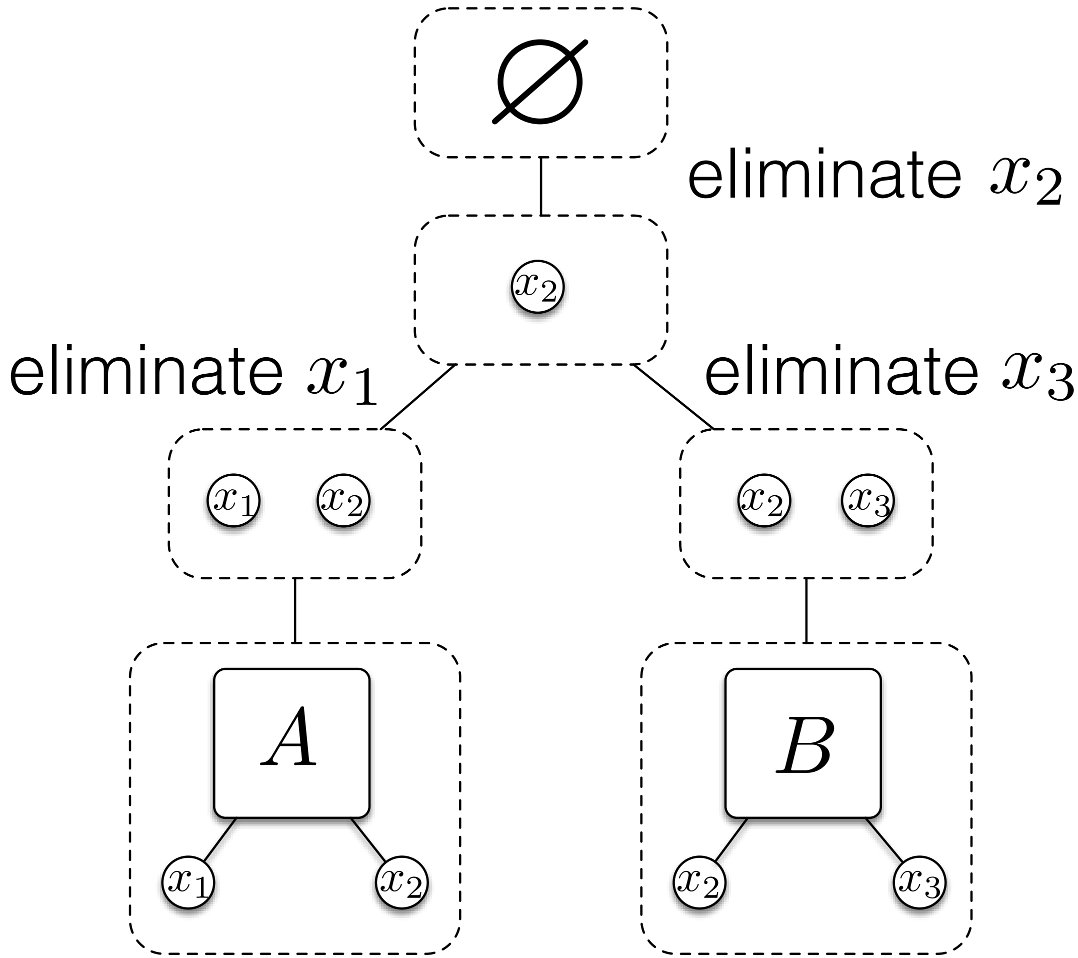
Graphical counterpart of hierarchical terms

$$(x_2)((x_1)A(x_1,x_2) \parallel (x_3)B(x_2,x_3))$$



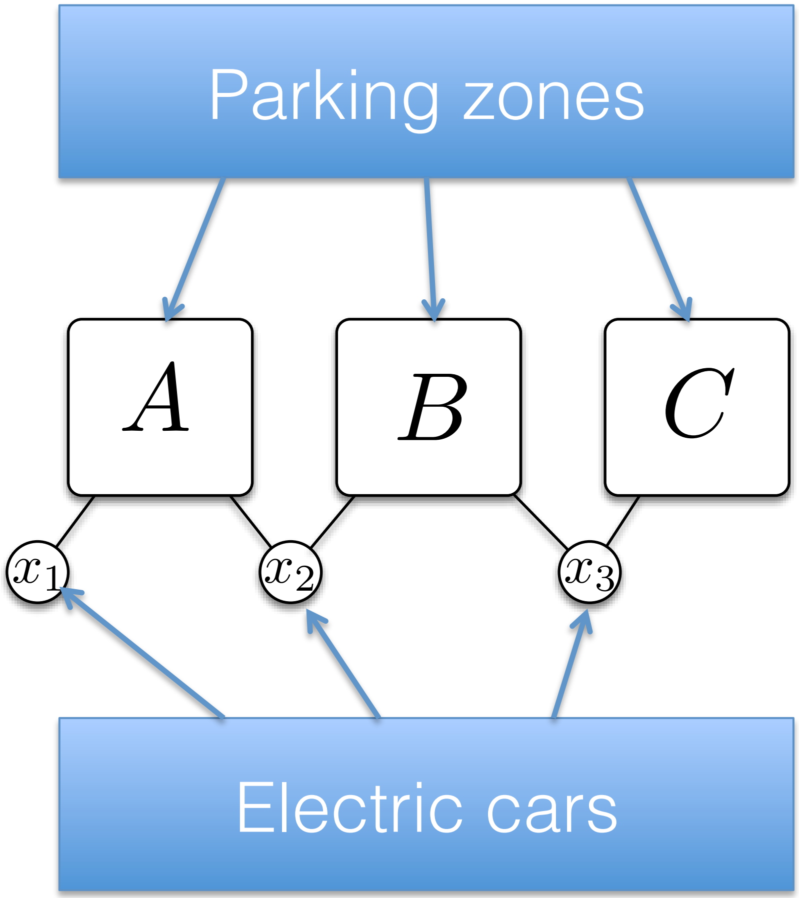
Dynamic programming algorithm

Bottom-up visit of the tree



Compute cost functions (tables) of leaves

ASCENS parking optimization problem



Goal: finding the best allocation of cars to parking zones

Algebraic representation

Boolean variable assignments

- "outside"

✓ "inside"

$$A(x_1, \ldots, x_n)$$

cars $x_1, \ldots, x_n$ may be parked inside *A*

$$(x)p$$

x must be parked inside of p

A “more efficient” algebra of cost functions

$$[\![p]\!]^c : (\mathcal{N} \to \{-, \checkmark\}) \to \mathbb{R}_\infty$$

A "more efficient" algebra of cost functions

$$[\![p]\!]^c : (\mathcal{N} \to \{-, \checkmark\}) \to \mathbb{R}_\infty$$

$$X \subseteq fn(p)$$

(other variables do not matter)

A "more efficient" algebra of cost functions

$$[\![p]\!]^c : (\mathcal{N} \to \{-, \checkmark\}) \to \mathbb{R}_\infty$$

$$[\![p \parallel q]\!]^c X = \min_{\{X_1, X_2\} \in \mathcal{P}_2(X)} \left\{ [\![p]\!]^c X_1 + [\![q]\!]^c X_2 \;\middle|\; \begin{array}{l} X_1 \subseteq fn(p), \\ X_2 \subseteq fn(q) \end{array} \right\}$$

$$[\![(x)p]\!]^c X = [\![p]\!]^c (X \cup \{x\})$$

Conclusions

Algebraic specification for optimization problems

- Variable elimination as variable binding
- Correctly handled via permutation algebras

Flat	Hierarchical (no scope extension)
Nominal terms/graphs	Hierarchical terms/graphs
Recursive computation of cost functions	Dynamic Programming algorithm

Future work

- Heuristics
- Dynamic graphs
- Precise correspondence between class of terms and nominal graphs
- Formalizing nominal structure