Reformulations in Mathematical Programming

Leo Liberti

LIX, École Polytechnique, France

Mathematical Programming

- Mathematical programs consist of sets of *parameters*, *variables*, *objective functions*, *constraints*
- The **objective functions** are mathematical expressions in terms of parameters and variables, together with an optimization direction
- The **constraints** are relations between mathematical expressions in terms of parameters and variables
- All such entities can also be expressed in terms of **indices**, which must be *quantified* over specified sets

The Language

- Consider an alphabet L including numbers, mathematical operators ($+, -, \times, \div, \uparrow, \sum, \prod, \log, \exp, \forall$), brackets, and symbols denoting parameters and variables
- **Given a sequence of elements of L, is it a well-formed statement of a mathematical program?**
- I.e. , we treat Mathematical Programming (MP) as a language, whose semantic purpose is to describe a set of points in a Euclidean space (the *optima*)
- One possible grammar of the MP language is specified in the appendix of the AMPL book

Main motivation

- Given an optimization problem, many different *MP formulations* can describe its solution set
- The performances of solution algorithms depend on the MP formulation
- **Given an optimization problem and a solution algorithm, what is the MP formulation yielding the best performance?**
- *How do we pass from one formulation to another that keeps some (all) of the mathematical properties of the old formulation?*

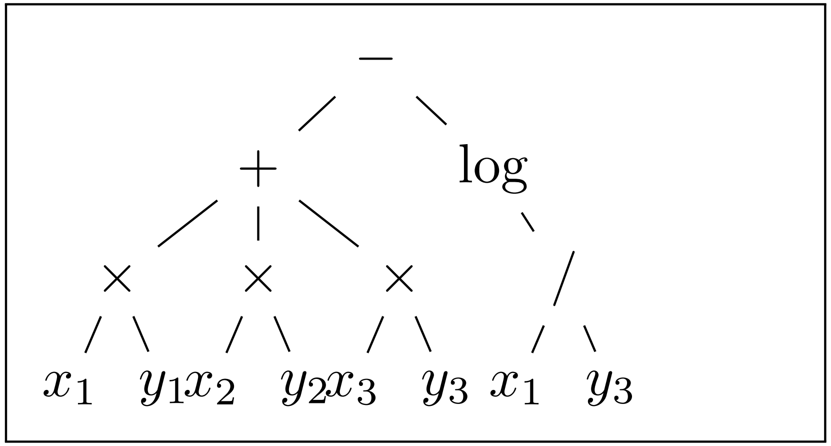
Reformulations: Existing definitions

- "Q is a reformulation of P": what does it mean?
- Definition in Mathematical Programming Glossary:
  Obtaining a new formulation Q of a problem P that is in some sense better, but equivalent to a given formulation. **Trouble:** vague.
- Definition by H. Sherali [private communication]:
  bijection between feasible sets, objective function of Q is a monotonic univariate function of that of P. **Trouble:** feasible sets bijection: condition is too restrictive
- Definition by P. Hansen [Audet et al., JOTA 1997]: P, Q *opt. problems; given an instance p of P and q of Q and an optimal solution y^* of q, Q is a reformulation of P if an optimal solution x^* of p can be computed from y^* within a polynomial amount of time*. **Trouble:** only maintains optimality, requires polynomial-time transformation

Storing MP formulations

- Mathematical expressions as n-ary *expression trees*

$$\sum_{i=1}^{3} x_i y_i - \log(x_1/y_3)$$



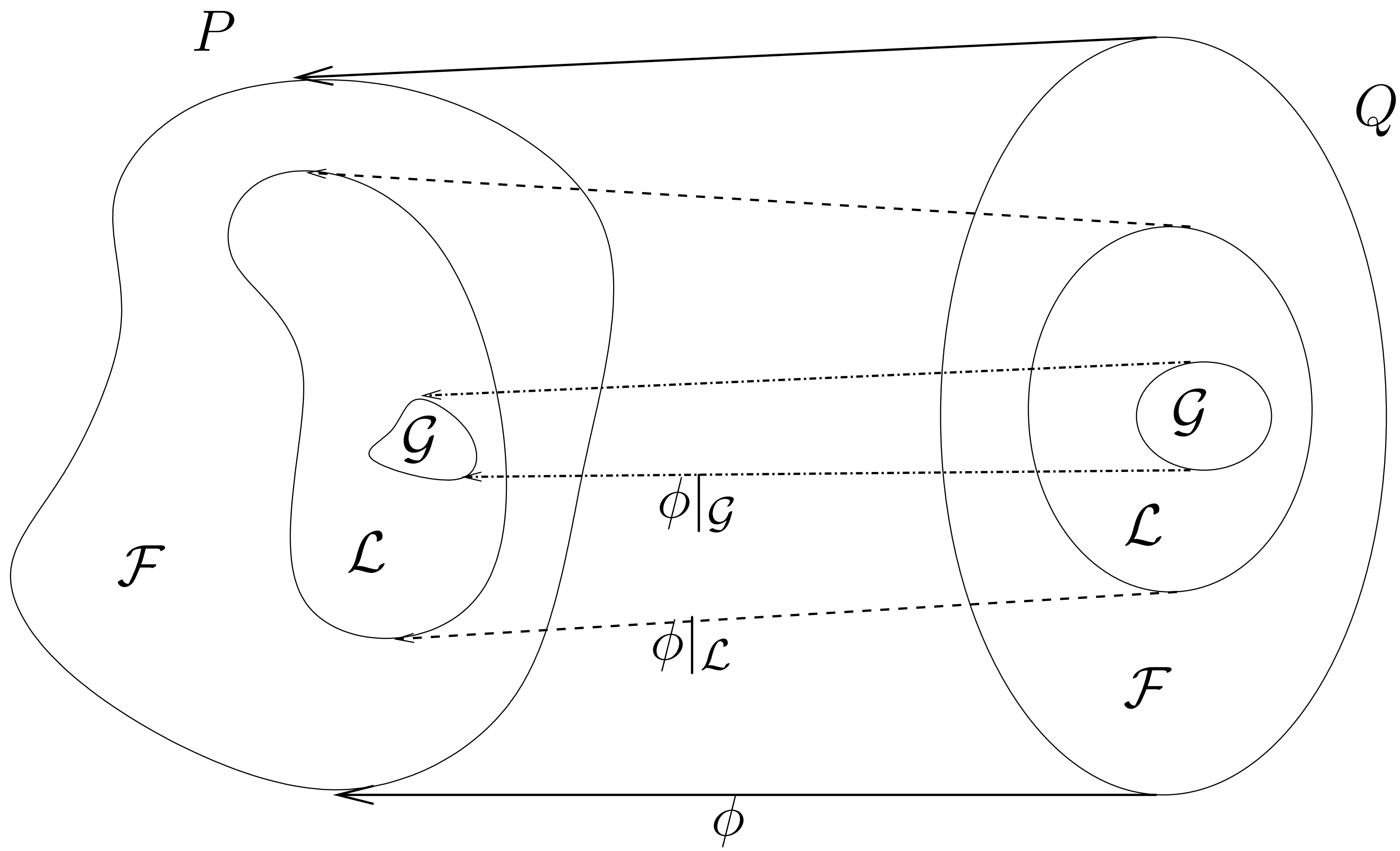
- A *formulation* P is a 7-tuple $(\mathcal{P}, \mathcal{V}, \mathcal{E}, \mathcal{O}, \mathcal{C}, \mathcal{B}, \mathcal{T})$ =(parameters, variables, expression trees, objective functions, constraints, bounds on variables, variable types)
- Objectives are encoded as pairs (d, f) where $d \in \{-1, 1\}$ is the optimization direction and f is the function being optimized
- Constraints are encoded as triplets $c \equiv (e, s, b)$ ($e \in \mathcal{E}$, $s \in \{\leq, \geq, =\}$, $b \in \mathbb{R}$)
- $\mathcal{F}(P)$ = feasible set, $\mathcal{L}(P)$ = local optima, $\mathcal{G}(P)$ = global optima

Auxiliary problems

If problems P, Q are related by a computable function f through the relation $f(P, Q) = 0$, Q is an *auxiliary problem* with respect to P.

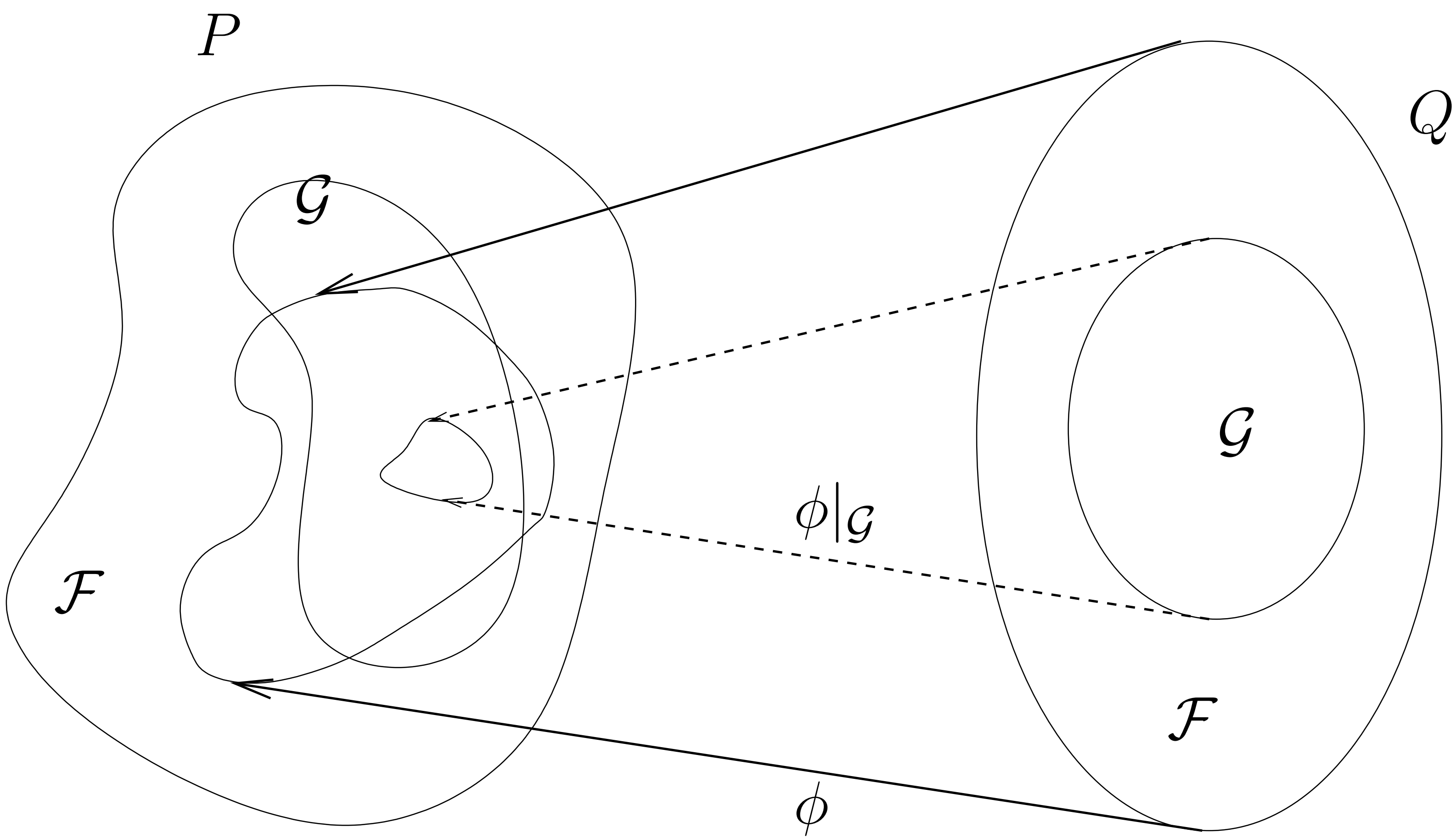
- **Opt-reformulations** (or *exact reformulations*): preserve all optimality properties
- **Narrowings**: preserve some optimality properties
- **Relaxations**: provide bounds to an objective function value towards its optimization direction
- **Approximations**: formulation Q depending on a parameter k such that "$\lim\limits_{k\to\infty} Q(k)$" is an opt-reformulation, narrowing or relaxation

Opt-reformulations



Main idea: if we find an optimum of Q, we can map it back to the same type of optimum of P, and for all optima of P, there is a corresponding optimum in Q.

Narrowings



Main idea: if we find a global optimum of Q, we can map it back to a global optimum of P. There may be optima of P without a corresponding optimum in Q.

Relaxations

- A problem Q is a *relaxation* of P if: (a) $\mathcal{F}(P) \subseteq \mathcal{F}(Q)$ and (b) for all $(f,d) \in \mathcal{O}(P)$, $(\bar{f},\bar{d}) \in \mathcal{O}(Q)$ and $x \in \mathcal{F}(P)$ we have $\bar{d}\bar{f}(x) \geq df(x)$
- **Relaxations guarantee the bound of all objectives over all the feasible region**

Approximations

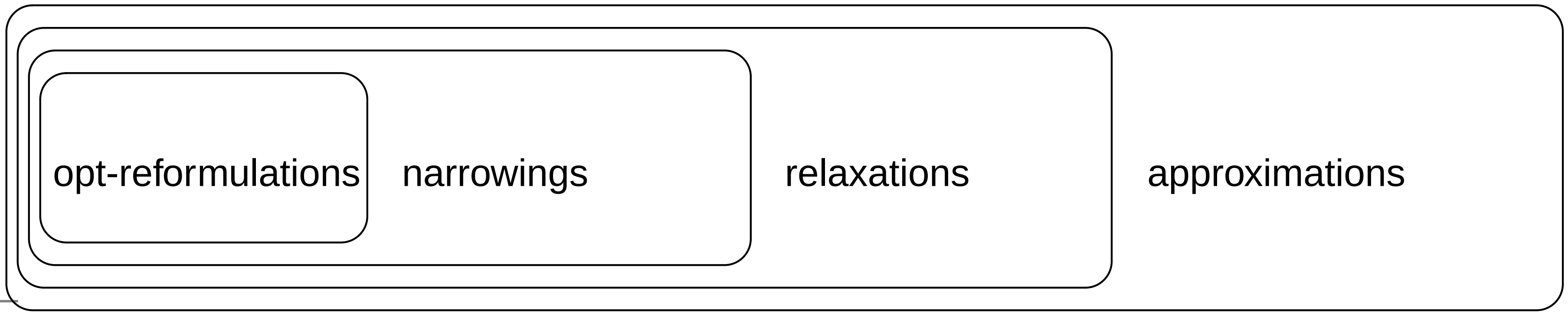
Q is an *approximation* of P if there exist: (a) an auxiliary problem Q^* of P; (b) a sequence $\{Q_k\}$ of problems; (c) an integer $k' > 0$; such that:

1. $Q = Q_{k'}$
2. $\forall f^* \in \mathcal{O}(Q^*)$ there is a sequence of functions $f_k \in \mathcal{O}(Q_k)$ converging uniformly to f^*;
3. $\forall c^* = (e^*, s^*, b^*) \in \mathcal{C}(Q^*)$ there is a sequence of constraints $c_k = (e_k, s_k, b_k) \in \mathcal{C}(Q_k)$ such that e_k converges uniformly to e^*, $s_k = s^*$ for all k, and b_k converges to b^*.

There can be approximations to opt-reformulations, narrowings, relaxations

Composition laws

- Opt-reformulation, narrowing, relaxation, approximation are all transitive relations, **so they can be chained**
- An approximation of *any reformulation chain* is an approximation
- A reformulation chain involving *opt-reformulations, narrowings, relaxations* is a relaxation
- A reformulation chain involving *opt-reformulations and narrowings* is a narrowing
- A reformulation chain involving *opt-reformulations only* is an opt-reformulation



Research programme

- Identify a library of reformulations that can be carried out *automatically* (by a computer) — **under way**
- Implement data structures for holding MP formulations as well as algorithms for changing their structures — **under way**
- Create a language for combining elementary reformulations into complex (possibly conditional) reformulations, according to the composition laws above — **to do**
- Create a heuristic method for finding out the *best* reformulation given an optimization problem and a solution algorithm — **to do**