Cutting plane and column generation methods using interior point methods

John E. Mitchell

Mathematical Sciences,
Rensselaer Polytechnic Institute,
Troy NY 12180 USA

Visiting: TWI/SSOR,
Delft University of Technology,
2628 CD Delft,
The Netherlands

email: mitchj@rpi.edu or J.E.Mitchell@twi.tudelft.nl
http://www.math.rpi.edu/~mitchj

TWI/SSOR, TU Delft
25 November, 1997

Abstract

Column generation methods can be used to solve integer programming problems, stochastic programming problems, crew scheduling problems, and multicommodity network flow problems, among others. These methods solve a sequence of linear programming subproblems. The rapid increase in computational power in the last few years has made it possible to solve very large instances. For such problems, it is often more efficient to use an interior point method to solve the linear programming subproblems than to use the simplex method. We survey both the theoretical and computational research in this area.

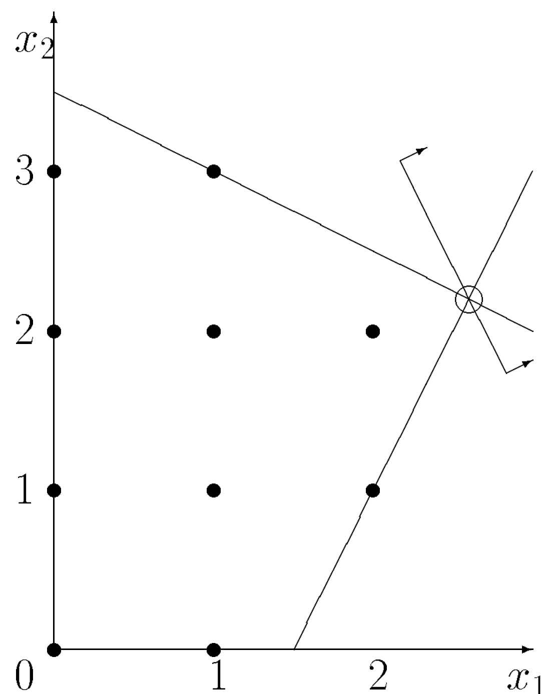
Introduction

- Interested in linear programming problems with a **large number of constraints** or a **large number of variables**.
- Solve problems using **constraint generation** or **column generation** methods.
- Solve linear programming subproblems using **interior point methods**
- These techniques are used to solve:
  - Integer programming problems
  - Multicommodity network flow problems
  - Stochastic programming problems
  - Semi-infinite programming problems
  - Airline crew scheduling problems
  - Bounded error parameter estimation
  - Adaptive filtering

Solving an integer program with cutting planes

Example:

$$
\begin{array}{lrcrcl}
\min & -2x_1 & - & x_2 & & \\
\text{s.t.} & x_1 & + & 2x_2 & \leq & 7 \\
& 2x_1 & - & x_2 & \leq & 3 \\
& & & x_1, x_2 & \geq & 0, \text{ integer}
\end{array}
$$

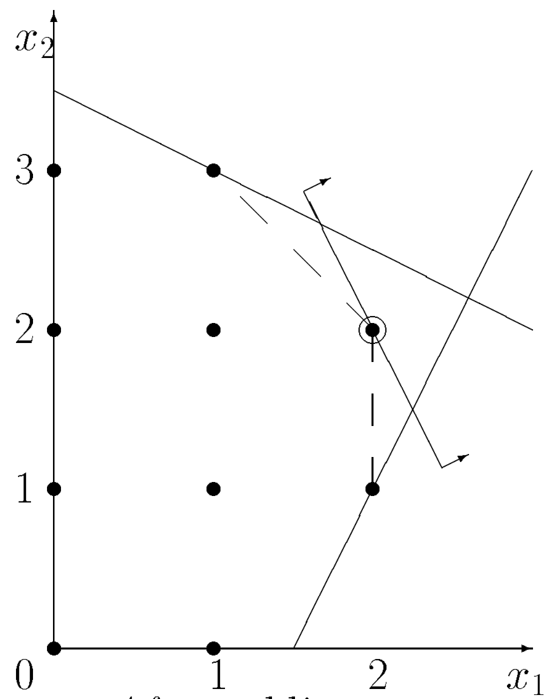


Linear programming relaxation

Solve LP relaxation.
Obtain $x_1 = 2.6$, $x_2 = 2.2$, value -7.4.

Cutting planes:

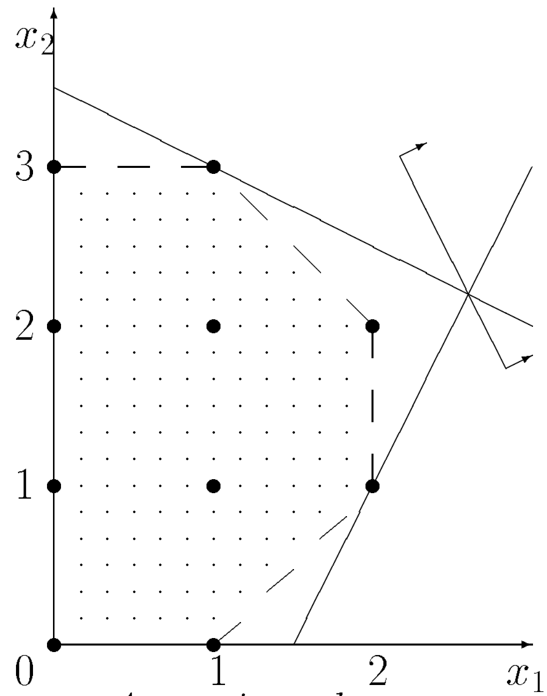
$$\begin{array}{lrcrcl} \min & -2x_1 & - & x_2 & & \\ \text{s.t.} & x_1 & + & 2x_2 & \leq & 7 \\ & 2x_1 & - & x_2 & \leq & 3 \\ & \boldsymbol{x_1} & + & \boldsymbol{x_2} & \leq & \mathbf{4} \\ & \boldsymbol{x_1} & & & \leq & \mathbf{2} \\ & & & x_1, x_2 & \geq & 0 \end{array}$$



After adding two cuts

- Solve. Optimal solution is $x_1 = 2$, $x_2 = 2$, value -6.
- Note: **feasible** in the original integer program, so **optimal**, since optimal for a *relaxation*.

Convex hull of integer points:



A cutting plane example

If we knew all the constraints of the convex hull of the integer program then we could solve the integer program by solving one linear program. Unfortunately, hard to get such a description.

Basic structure of a cutting plane algorithm

1. **Solve** the linear programming **relaxation**.
2. If the solution to the relaxation is feasible in the integer programming problem, **STOP** with optimality.
3. Else, find one or more cutting planes that **separate** the optimal solution to the relaxation from the convex hull of feasible integral points, and add a subset of these constraints to the relaxation.
4. Return to step 1.

Notes:

- Classically, first relaxation is solved using the primal simplex algorithm.
- After the addition of cutting planes, the current **primal** iterate becomes **infeasible**.
- However, the change to the dual problem is the addition of some columns with corresponding variables.
- If these extra dual variables are given the value 0 then the current dual solution is still **dual feasible**.
- Therefore, subsequent relaxations are solved using the dual simplex method in the classical approach. Dual simplex method is quick to reoptimize after addition of just a few constraints.
- Notice that the values of the relaxations provide **lower bounds** on the optimal value of the integer program. These lower bounds can be used to measure progress towards optimality, and to give performance guarantees on integral solutions.

Interior point methods for linear programs

- Interior point methods generally faster than simplex for large problems, ie, more than a thousand variables and/or constraints.
- Interior point methods try to keep iterates **well-centered**: good to keep $x_i z_i \approx \mu$ for all components, where μ varies from iteration to iteration. (z=dual slack.) If $x_i z_i = \mu$ for all components then the point is on the **central trajectory**.
- Duality gap is approximately $n\mu$. (n is number of primal variables.)
- Harder to exploit a **warm start** with an interior point method.

A closer look at adding a constraint

Approximately solve primal-dual LP pair

$$\begin{array}{llll} \min & c^T x & & \\ \text{s.t.} & Ax & = & b \quad (P) \\ & x & \geq & 0 \end{array} \qquad \begin{array}{llll} \max & b^T y & & \\ \text{s.t.} & A^T y & \leq & c \quad (D) \end{array}$$

- Approximate solution: x^*, y^*.
- Use interior point method so $x^* > 0$, $A^T y < c$.
- Decide to add constraint $g^T x \geq h$.

Modified primal-dual pair:

$$\begin{array}{llllll} \min & c^T x & & & & \\ \text{s.t.} & Ax & & & = & b \quad (P_0) \\ & \boldsymbol{g^T x} & - & \boldsymbol{x_0} & = & \boldsymbol{h} \\ & & & x, \boldsymbol{x_0} & \geq & 0 \end{array}$$

$$\begin{array}{llllll} \max & b^T y & + & \boldsymbol{h y_0} & & \\ \text{s.t.} & A^T y & + & \boldsymbol{g y_0} & \leq & c \quad (D_0) \\ & & & \boldsymbol{y_0} & \geq & 0 \end{array}$$

- Now $\boldsymbol{g^T x^* < h}$, since cutting plane. So infeasible in (P_0).
- In (D_0), $y = y^*$, $y_0 = 0$ is feasible.

Restarting with interior point method:

- Have dual feasibility, but not **interior** dual point.
- Use **affine direction** to regain dual interior point:

$$\Delta y = -(AA^T)^{-1}Ag, \qquad \Delta y_0 = 1.$$

- This direction gives interior point with minimal disruption to dual feasibility. Adjustment to $A^T y + g y_0$ is projection of g onto null space of A.
- Not easy to regain **primal** interior feasible point. Options:
  - Infeasible primal-dual method.
  - Backup to earlier iterate.
  - Dual method.
- Would also like to get a well-centered iterate.

Analytic Center Cutting Plane Method (ACCPM)

(Goffin, Vial *et al.*, late 80's onwards)

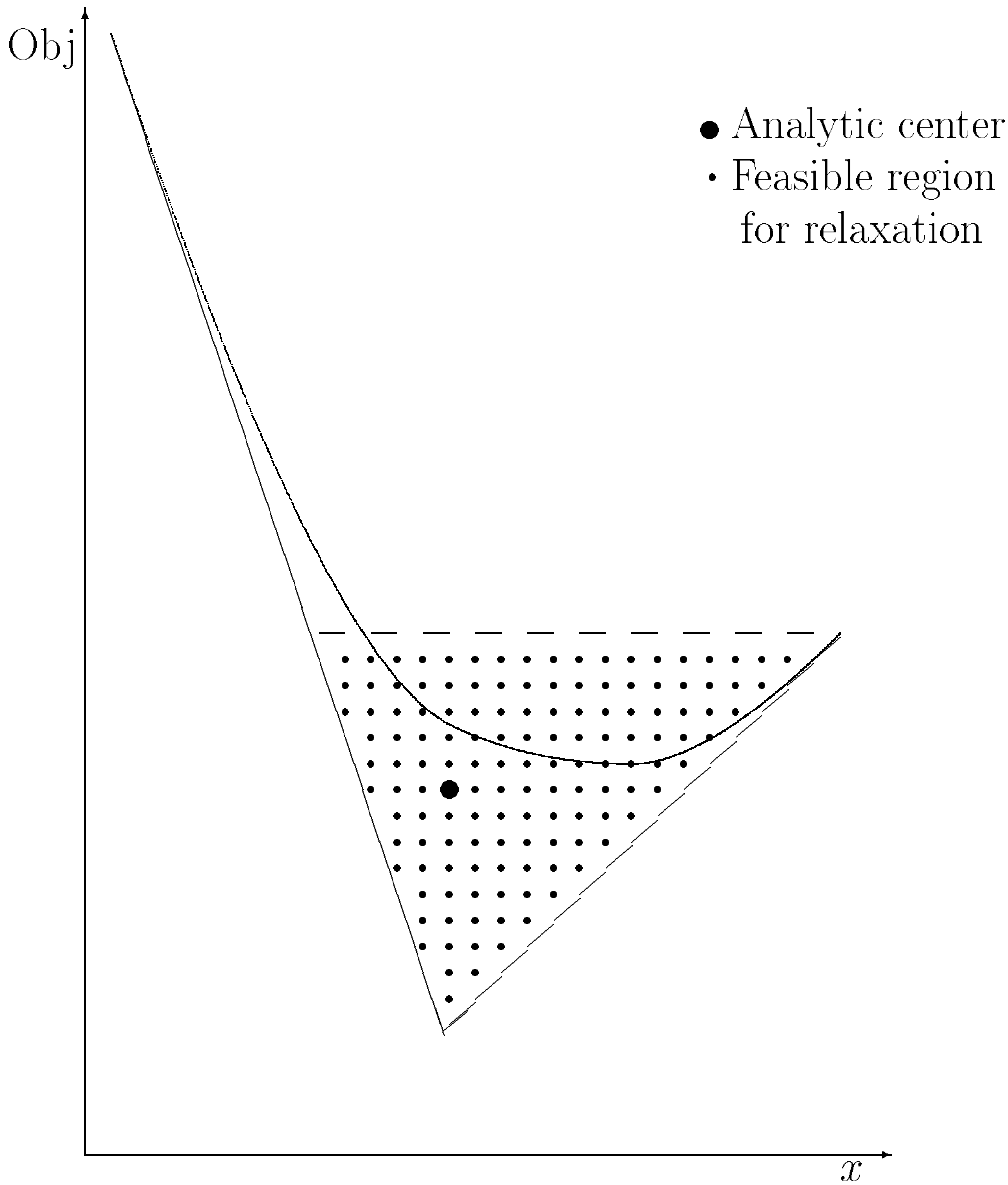
- Originally developed for nondifferentiable optimization problems. (Goffin, Haurie, Vial (1992).)
- Subsequently applied to stochastic programming problems, multicommodity network flow problems.
- Similar to the (independently developed) Mitchell-Todd (1992) application of the primal projective algorithm to solve integer programming problems with an interior point cutting plane method.

How the algorithm works

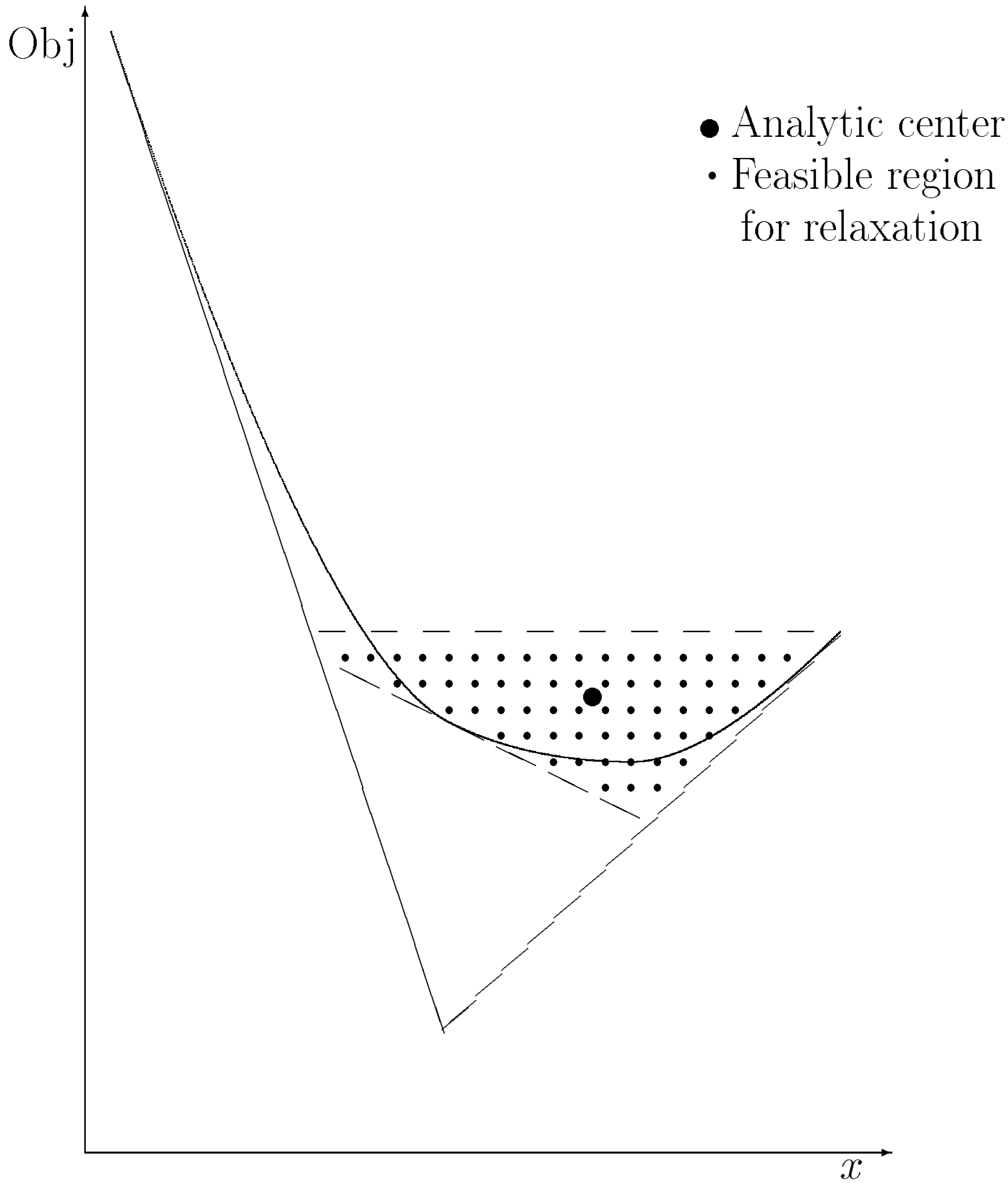
- Find a weighted analytic center of an approximation to the set of optimal solutions.
- If this center is feasible, use a contour of the objective function for a cut.
- Else, cut off the center.
- Return to the first step.

Example of ACCPM

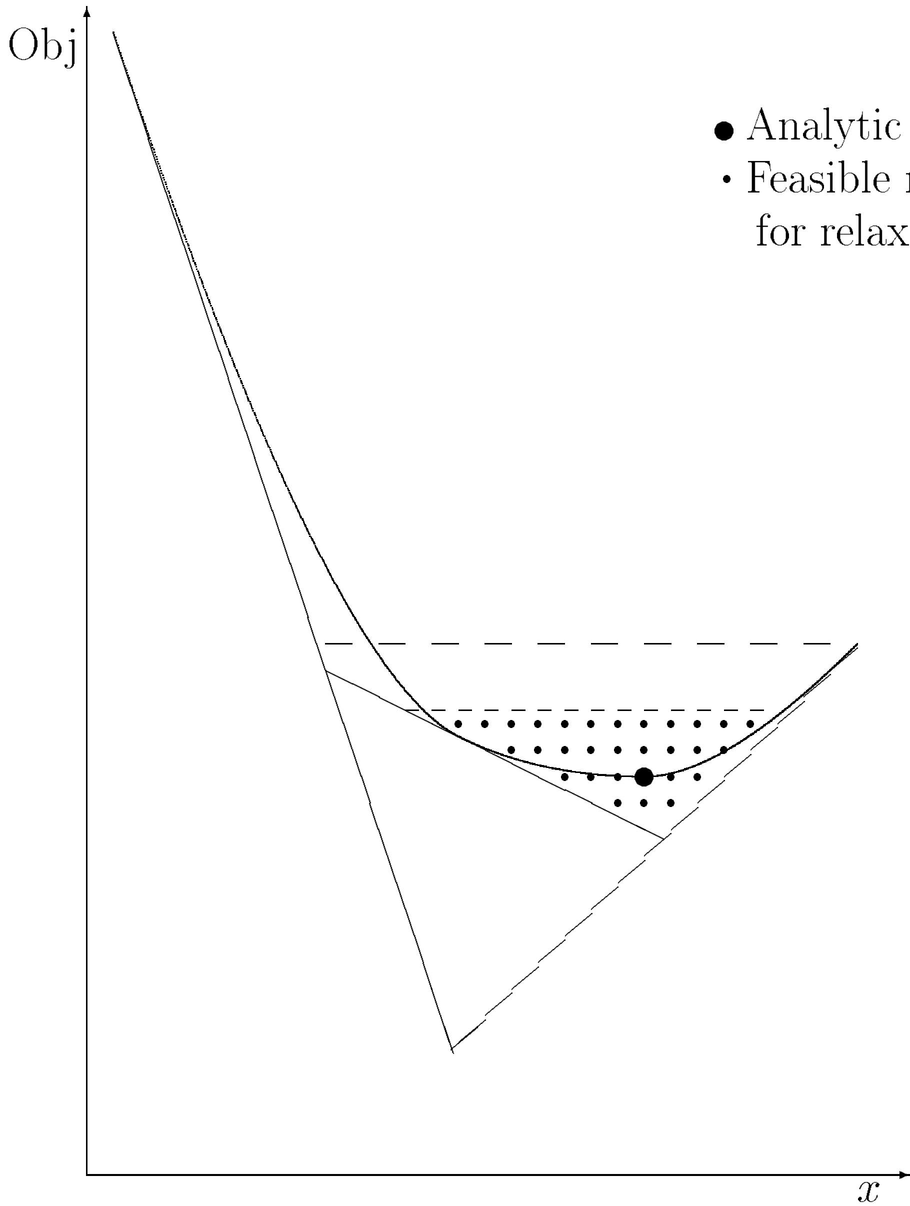
- Want to find lowest point on curve.
- Use piecewise linear approximation.



Analytic center infeasible, so cut it off:



Analytic center feasible, so cut on objective function:



Analytic center optimal for original problem, so STOP.

Features of Algorithm ACCPM:

- The pictures correspond to the primal problem:

$$\begin{array}{ll} \min & c^T x \\ \text{s.t.} & Ax \geq b \end{array}$$

- Have an upper bound UB on optimal value.
- So have a **set of localization**

$$L := \{x : Ax \geq b, c^T x \leq UB\}.$$

- Corresponding dual problem is

$$\begin{array}{lrcl} \max & b^T y & & \\ \text{s.t.} & A^T y & = & c \\ & y & \geq & 0 \end{array}$$

- The algorithm works in the dual space, generating a primal solution once close to the analytic center.
- Typically, only spend **five or six** iterations on each subproblem.
- Any feasible dual solution gives a lower bound LB on the optimal value of the underlying problem.

The algorithm itself:

1. Have a **set of localization** for the primal problem, and upper and lower bounds $\boldsymbol{UB}$ and $\boldsymbol{LB}$.
2. Approximately minimize the **potential function** $\boldsymbol{\psi(y, UB) := (m+1)\log(UB - b^T y) - \Sigma_{j=1}^{m} \log(y_j)}$.
3. Once the gradient is small enough, **generate a primal solution**. The primal and dual solutions are an approximate primal-dual analytic center.
4. Search for cutting planes:
   - If the primal point is infeasible in the relaxation, **improve the approximation** to the function.
   - If the primal point is feasible, **improve the upper bound $\boldsymbol{UB}$**.
5. If the gap between UB and LB small enough, **STOP**.
6. Else, **loop**: return to step 2.

- **Note:** The bound in the potential function is not updated while the relaxation is being solved.

Using ACCPM to solve stochastic programs:

- Bahn, Du Merle, Goffin, Vial (1995)
- **Two-stage stochastic program with recourse** given by

$$\begin{array}{c} \min\ c_0^T x_0 + \int_{\xi\in\Xi} Q(x_0,\xi) \\ \text{s.t. } A_0 x_0 \geq b_0 \end{array}$$

- with recourse function

$$Q(x_0,\xi) := \inf_x \{c(\xi)^T x \mid B(\xi)x \geq b(\xi) - A(\xi)x_0\}$$

- Get a set of constraints for each realization ξ of the **uncertainty**. These constraints are added as cutting planes.
- Solve problems with several thousand rows and variables.
- Considerably faster than solving the full LP. Also faster than Dantzig-Wolfe on some problems.
- ACCPM has also been successfully applied to **multicommodity network flow** problems and **nondifferentiable optimization** problems (Goffin, Gondzio, Sarkissian, Vial (1997), Goffin, Haurie, Vial (1992)).

Integer programming problems

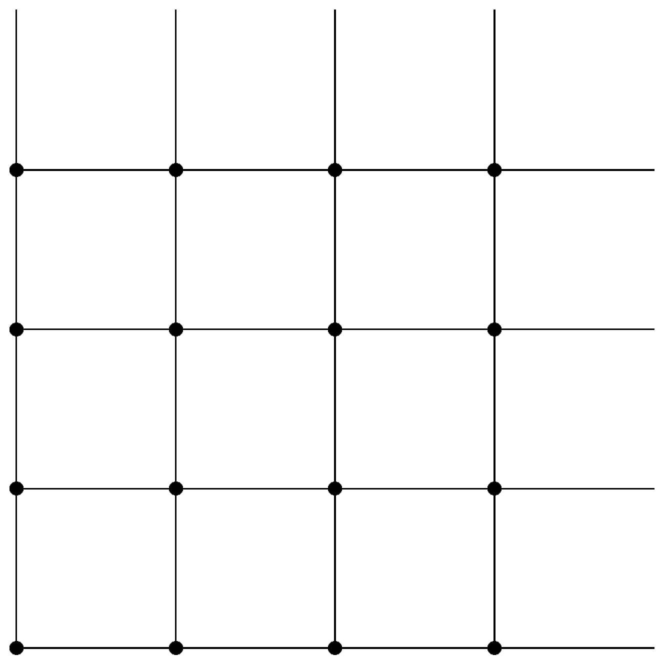
- Mitchell (1997a, 1997b), Mitchell and Borchers (1996, 1997) used **primal-dual** interior point cutting plane methods.
- Solve LP relaxations **approximately**.
- **Primal heuristics:** Round the interior iterate to give a good integer feasible solution.
- Use dynamically altered tolerance on duality gap to decide how accurately to solve relaxations.
- Also check primal and dual values against best heuristic value when deciding whether to search for cutting planes.
- Restart by backing up towards a **known primal feasible** point. For example, for maximum cut problem, the point with every component equal to 0.5 is feasible. Thus, restart at convex combination of this point and final iterate, typically 95% of way to boundary.
- Restart dual by returning to an earlier iterate.
- Any variable too close to its bounds is set to 10^{-4}, say.

The ground state of an Ising Spin Glass

- Problem in glassy dynamics in statistical physics
- An *Ising spin glass* is a model of a magnetic material, and it consists of a grid of magnetic spins.
- Each spin S_i is in one of two states, which we call "up" and "down"; we assign S_i the value $+1$ if the spin is up and -1 if the spin is down.
- We assume the grid is an $L \times L$ square grid embedded on a torus.
- Further, we assume that the interactions between spins are restricted to neighbours, that is, we consider the short range model with Ising spins S_i.

Graph representation:

4×4 grid on a torus:



- Know **node interactions**, ± 1. These are edge weights.
- Find **state** of each vertex.
- If edge between neighbouring vertices has value $J_{ij} = +1$, want vertices to be in *same* state.
- If edge between neighbouring vertices has value $J_{ij} = -1$, want vertices to be in *opposite* states.

Integer programming model

- Minimize the Hamiltonian of the energy:

$$H := -\sum_{\text{neighbours } i,j} J_{ij} S_i S_j$$

  with $S_i = \pm 1$.

- Can be modelled as a **Max Cut** problem: all the *up* vertices on one side of the cut, and all the *down* vertices on the other side, with edge weights derived from J_{ij}.

- So solve

$$\min\{c^T x : x \text{ is the incidence vector of a cut}\}$$

  Here, x has one component for each *edge*. The optimal value of this problem is *even*.

Polyhedral theory:

- **Initial relaxation:**

$$\min\{c^T x : 0 \leq x_e \leq 1\}$$

- **Cutting planes from cycles:** Every cycle and every cut intersect in an even number of edges. Gives the following facet defining inequality for every subset F of odd cardinality of every chordless cycle C:

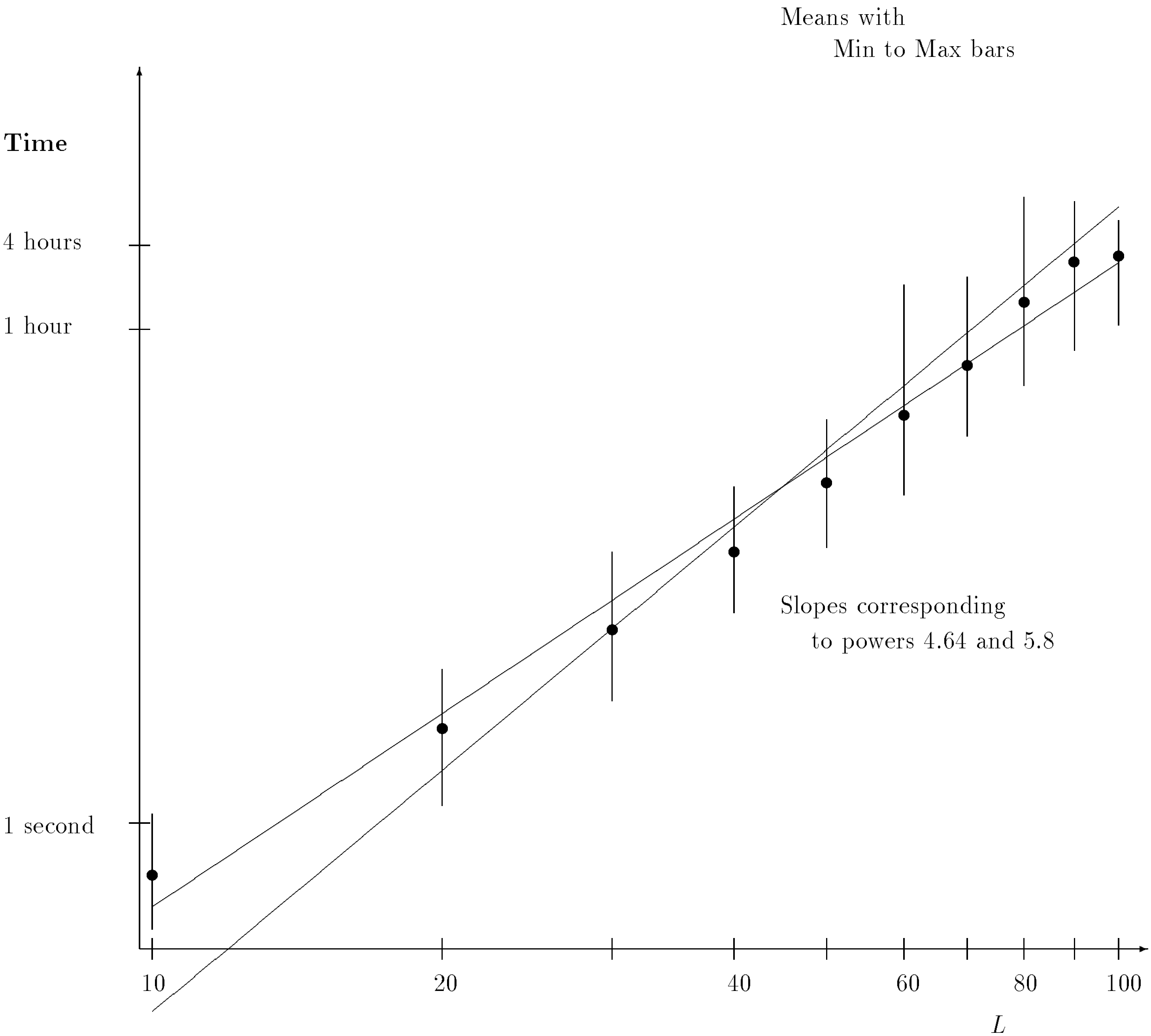
$$x(F) - x(C \setminus F) \leq |F| - 1$$

- All the **squares** in the grid are chordless cycles. There are also many other chordless cycles.

- Any integral vector that satisfies all the cycle inequalities must be the incidence vector of a cut.

- There are also other families of valid inequalities.

Run times


Means with
Min to Max bars
Time
4 hours
1 hour
1 second
Slopes corresponding
to powers 4.64 and 5.8
10
20
40
60
80
100
L

Results

- Solved 100 problems of each size 10×10, 20×20, ..., 100×100.
- Problems of size 100×100 are solved in an average of **3hrs, 20 minutes** on a Sun SPARCstation 20/71. By comparison, a simplex cutting plane code using CPLEX 3.0 on a Sun SPARCstation 10 required up to a **day** to solve problems of size 70×70. (De Simone *et al*, 1996.)
- Fitting $\log(Time)$ versus $\log(L)$ gives a slope of 4.64. This shows that runtime only grows at rate $L^{4.64}$. By comparison, the simplex runtimes appeared to grow at a rate of L^6.
- The number of iterations per linear program averages out to around 8. Fewer iterations are required in earlier relaxations and more iterations for later relaxations.
- One possible way to improve the algorithm would be to **crossover** from an interior point method to a simplex method after a certain number of stages.

The linear ordering problem
(Mitchell and Borchers, 1997.)

- **Applications** include
  - Triangulation of input-output matrices in economics
  - Archeological seriation
  - Minimizing total weighted completion time in one-machine scheduling
  - Aggregation of individual preferences
- **Example**: comparing Dutch cheeses:
  Each of a group of people perform *pairwise comparisons* between the cheeses. **Objective:** determine the overall preference order for the group.
- **In general**:
  - Have p objects to place in order.
  - If place i before j, pay cost $g(i, j)$.
  - Conversely, if place i after j, pay cost $g(j, i)$.
  - Choose ordering to minimize the cost.

Modelling the problem:

- **Variables:** Define

$$x(i,j) = \begin{cases} 1 & \text{if } i \text{ before } j \\ 0 & \text{otherwise} \end{cases}$$

- **Eliminate variables:**
Must have $x(i,j) + x(j,i) = 1$ for each pair $1 \leq i < j \leq p$. Use this to eliminate variables $x(j,i)$, $j > i$.

- **Objective:**
$x(i,j)$, $i < j$, has cost coefficient

$$c(i,j) := g(i,j) - g(j,i).$$

- **Initial relaxation:**

$$\begin{array}{lll} \min & \Sigma_{i=1}^{p-1} \Sigma_{j=i+1}^{p} c(i,j)x(i,j) & \\ \text{subject to} & 0 \leq x(i,j) \leq 1, & 1 \leq i < j \leq p \end{array}$$

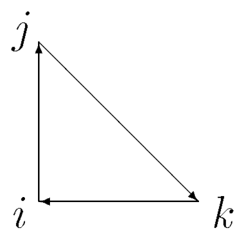
Cutting planes

- Use **triangle inequalities** to prevent

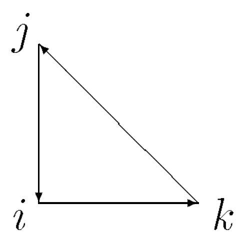
  i before j before k before i
  or: $x(i,j) = x(j,k) = x(k,i) = 1$

  Enforced by $x(i,j) + x(j,k) + x(k,i) \leq 2$.

- For $1 \leq i < j < k \leq p$, get two forms:



$$x(i,j) + x(j,k) - x(i,k) \leq 1$$



$$-x(i,j) - x(j,k) + x(i,k) \leq 0$$

- If x is integral and satisfies all the triangle inequalities then it **solves the linear ordering problem**.

- Other inequalities exist. We only used triangle inequalities.

Crossover

Three different algorithms:

1. Use **interior point method exclusively** to solve the relaxations.
2. Use the **simplex method exclusively** to solve the relaxations.
3. **Combine** the two methods: use the interior point method to solve the first few relaxations and use the simplex method to solve the remaining relaxations.

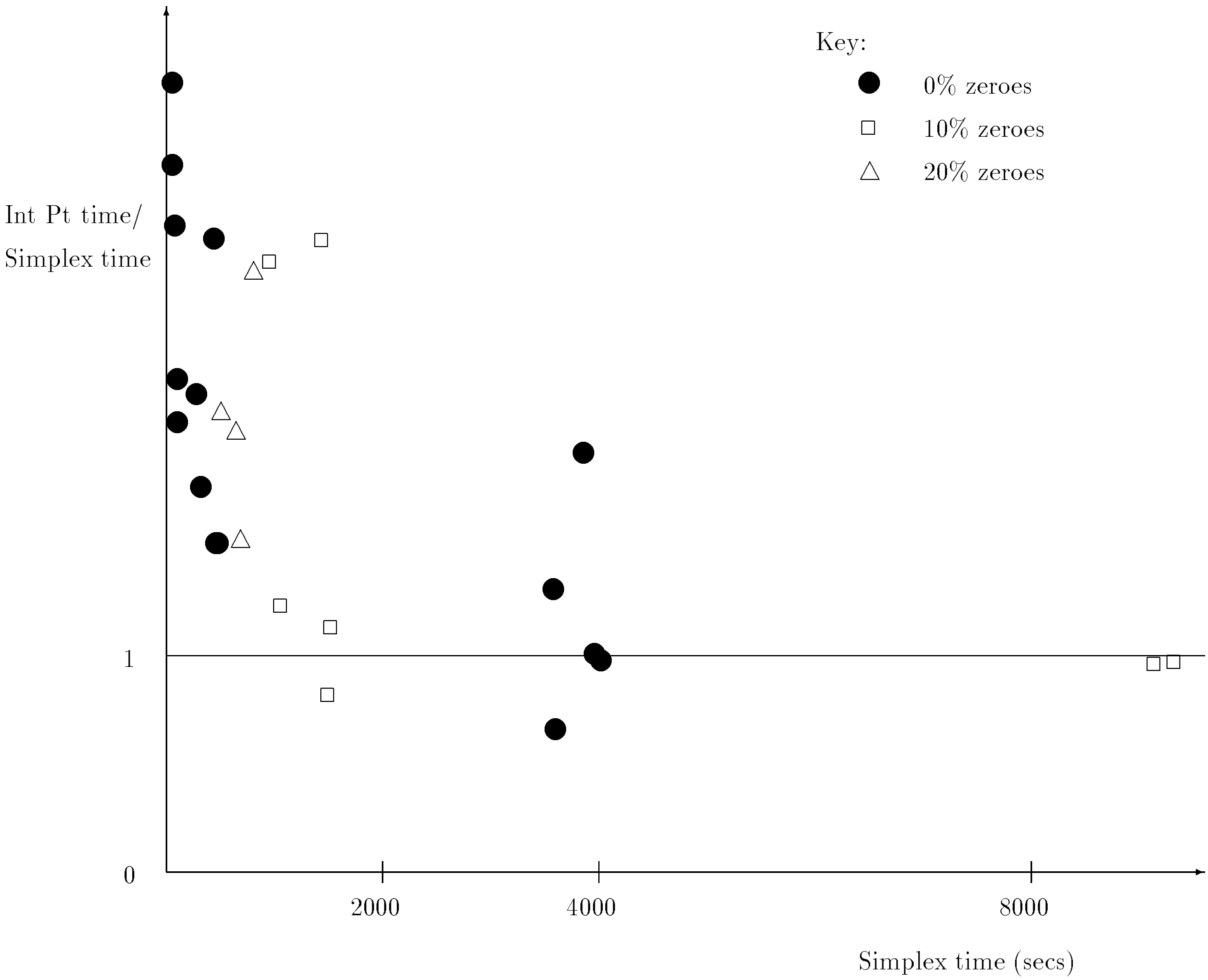
Experimented with different points to perform the crossover.

Randomly generated problems

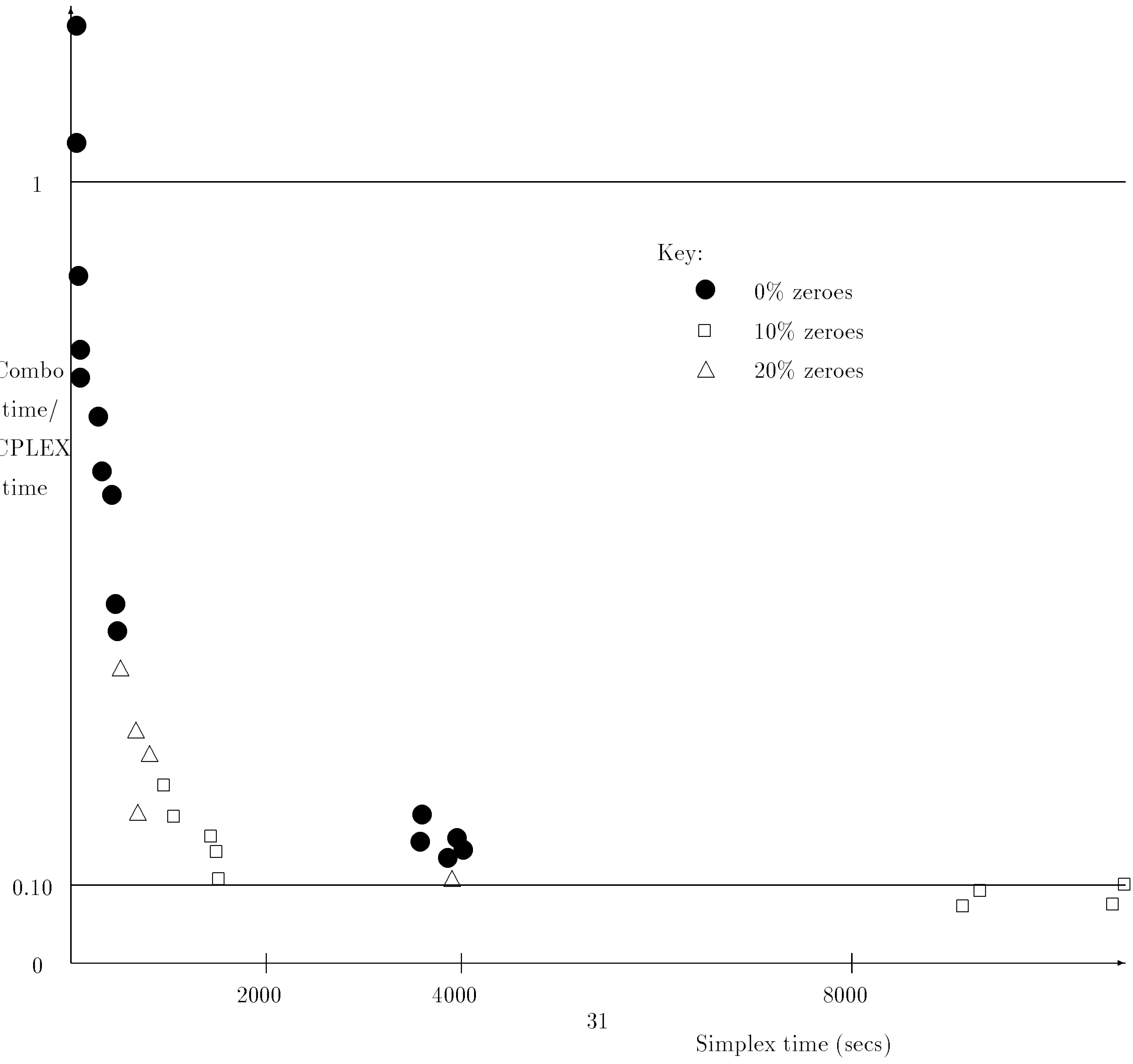
- Up to 250 objects.
- - For $i < j$, generate $g(i, j)$ uniformly between 0 and 99.
  - For $j < i$, generate $g(i, j)$ uniformly between 0 and 39.

  Randomly permute so the primal heuristic is not at an advantage.
- Zero out a percentage of the entries.
- Generated six different classes of problems. Five problems in each class. For each problem within a particular class, we used the same crossover criterion.
- Crossover points:
  - `r150.0` and `r200.0`: switch after two stages.
  - `r150.1` and `r250.0`: switch after three stages.
  - `r200.1`: switch after add $<$ 600 constraints in a stage. (On average, after 7 stages.)
  - `r100.2`: switch after add $<$ 100 constraints in a stage. (On average, after 7 stages.)

Time to solve with interior point code
vs time to solve with simplex code
(No crossover)



Combination code time vs Simplex time



Comments:

- For sufficiently hard problems, combining the two codes performs **significantly better** than either code individually.
- For larger problems, the interior point and simplex codes require comparable time. The interior point solver is a research code, and we believe based on our experience in solving standard test problems that this interior point solver is roughly half as fast as current high quality interior point solvers.
- Each of the criteria resulted in improvement over pure simplex for almost every set of problems.
- Final number of constraints for the pure integer code are approximately:
  - `r100.2`: 3000 constraints.
  - `r150.0`: 4000 constraints.
  - `r150.1`: 5000 constraints.
  - `r200.0`: 6500 constraints.
  - `r200.1`: 8000 constraints.
  - `r250.0`: 10000 constraints.

Infeasible interior point methods

- Gondzio (1996) has investigated using an **infeasible** primal dual interior point method in a column generation setting.
- Important to restart from an **approximate** solution to the relaxation. Can solve to optimality but should back up to a point with only three digits of accuracy, say.
- Introduces infeasibilities in both primal and dual when restarting, but maintains **μ-complementarity**.
- Use **target following algorithm** to recover feasibility, so initially emphasize **reducing infeasibilities**.
- Once infeasibilities are sufficiently reduced, use infeasible primal-dual interior point method to get new approximate solution.
- Good results on multicommodity network flow problems and other problems (Gondzio and Sarkissian, 1996).

Other applications

- **Bounded error parameter estimation**, Ye *et al.*, 1997. Have $y_i = q_i^T\theta + v_i$, for input q_i, output y_i, noise v_i and parameter θ. Want to estimate θ. Set up as quadratic program, generate new columns with each new observation. Successfully used interior point column generation method, with guarantees on performance. Obtain more centered final solution than when using least squares approach.

- **Adaptive filtering**, Luo *et al.*, 1997. Quadratic model to minimize estimated error. Data available over time. As more data becomes available, the solution is improved using a column generation approach with an interior point method. Gives better results than recursive least squares for problems with low signal to noise ratio, and comparable results under less adverse conditions.

- **Semi infinite programming**, Kaliski *et al.*, 1996. Solve with finite approximation in cutting plane scheme, as in Den Hertog *et al.* (1995). Use logarithmic barrier method to solve subproblems. Implemented in parallel with good results.

Theoretical properties

- The **ellipsoid algorithm** can be used to solve a column generation problem in polynomial time if cutting planes can be generated in polynomial time. (Grötschel *et al.*, 1988.)
- An exactly analogous result has **not been proved** for an interior point method.
- Early results showed that various algorithms are polynomial in the number of added cuts. (Den Hertog *et al.* (1995), Mitchell (1994), etc).
- Goffin *et al.* (1996) showed that ACCPM is **fully polynomial**, ie, the number of iterations to get a duality gap of ϵ depends on a power of ϵ.
- Atkinson and Vaidya (1995) developed a **polynomial** algorithm that requires that unimportant constraints be **dropped**.
- Mitchell and Ramaswamy (1993) extended the feasibility algorithm of Atkinson and Vaidya to follow the central trajectory and solve the problem to optimality. Further, they showed that a **long step** variant had the same complexity.

Many cuts, stronger cuts

- All the results mentioned earlier are for when just **one weak cut** is added at a time, that is, it is easily satisfied by the current point.
- For the Atkinson and Vaidya approach, the results can be extended to the case where **many cuts** are added right through the current iterate (Ramaswamy and Mitchell, 1994).
- For algorithms of the form ACCPM, results have been proven for adding **many weak cuts**, or for adding a single cut through the **center** (Goffin *et al.* (1993), Luo 1997).
- Goffin and Vial (1996) showed that the ACCPM still converges even when a single cut that makes the current iterate **infeasible** is added.
- Goffin and Vial (1997) showed that the ACCPM is fully polynomial if **two cuts** are added through the current iterate.

Conclusions

Good for problems which are

- Large (thousands of constraints and/or variables)
- Add many constraints at once (at least tens, preferably hundreds or thousands)
- Only need to solve approximately.
- Also useful if the linear programs are degenerate.

Should be implemented to

- Only solve relaxations approximately.
- Restart with an emphasis on being centered.
- Expend effort to obtain good heuristic solution, so can exploit good bounds.
- Occasionally solve relaxations to optimality.

Theoretically:

- Best complexity comparable with ellipsoid method.
- Still no polynomial interior point cutting plane algorithm which does not need to drop unimportant constraints.
- ACCPM still to be analyzed with many cuts added at current point.

References

[1] K. H. Afkhamie, Z.-Q. Luo, and K. M. Wong. Interior point column generation algorithms for adaptive filtering. Technical report, Electrical and Computer Engineering, McMaster University, Hamilton, Ontario, L8S 4L7, Canada, June 1997.

[2] D. S. Atkinson and P. M. Vaidya. A cutting plane algorithm for convex programming that uses analytic centers. *Mathematical Programming*, 69:1–43, 1995.

[3] O. Bahn, O. Du Merle, J. L. Goffin, and J. P. Vial. A cutting plane method from analytic centers for stochastic programming. *Mathematical Programming*, 69:45–73, 1995.

[4] E.-W. Bai, Y. Ye, and R. Tempo. Bounded error parameter estimation: A sequential analytic center approach. Technical report, Department of Management Science, University of Iowa, Iowa City, Iowa 52242, February 1997.

[5] C. De Simone, M. Diehl, M. Jünger, P. Mutzel, G. Reinelt, and G. Rinaldi. Exact ground states of two-dimensional $\pm J$ Ising spin glasses. *Journal of Statistical Physics*, 84:1363–1371, 1996.

[6] J.-L. Goffin, J. Gondzio, R. Sarkissian, and J.-P. Vial. Solving nonlinear multicommodity network flow problems by the analytic center cutting plane method. *Mathematical Programming*, 76:131–154, 1997.

[7] J.-L. Goffin, A. Haurie, and J.-P. Vial. Decomposition and nondifferentiable optimization with the projective algorithm. *Management Science*, 38:284–302, 1992.

[8] J.-L. Goffin, Z.-Q. Luo, and Y. Ye. Further complexity analysis of a primal-dual column generation algorithm for convex or quasiconvex feasibility problems. Technical report, Faculty of Management, McGill University, Montréal, Québec, Canada, November 1993.

[9] J.-L. Goffin, Z.-Q. Luo, and Y. Ye. Complexity analysis of an interior cutting plane method for convex feasibility problems. *SIAM Journal on Optimization*, 6:638–652, 1996.

[10] J.-L. Goffin and J.-P. Vial. Cutting planes and column generation techniques with the projective algorithm. *Journal of Optimization Theory and Applications*, 65(3):409–429, 1990.

[11] J.-L. Goffin and J.-P. Vial. Shallow, deep and very deep cuts in the analytic center cutting plane method. Technical Report Logilab Technical Report 96.1, HEC/Management Studies, University of Geneva, Geneva, Switzerland, May 1996.

[12] J.-L. Goffin and J.-P. Vial. A two-cut approach in the analytic center cutting plane method. Technical Report Logilab Technical Report 97.6, Logilab, Management Studies, University of Geneva, Geneva, Switzerland, August 1997.

[13] J. Gondzio. Warm start of the primal-dual method applied in the cutting plane scheme. Technical report, Logilab, HEC Geneva, Section of Management Sciences, University of Geneva, 102 Bd Carl Vogt, CH-1211 Geneva 4, Switzerland, May 1996. Revised: April 1997.

[14] J. Gondzio and R. Sarkissian. Column generation with a primal-dual method. Technical report, Logilab, HEC Geneva, Section of Management Sciences, University of Geneva, 102 Bd Carl Vogt, CH-1211 Geneva 4, Switzerland, June 1996.

[15] M. Grötschel, L. Lovasz, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag, Berlin, Germany, 1988.

[16] D. den Hertog, J. Kaliski, C. Roos, and T. Terlaky. A logarithmic barrier cutting plane method for convex programming problems. *Annals of Operations Research*, 58:69–98, 1995.

[17] J. Kaliski, D. Haglin, C. Roos, and T. Terlaky. Logarithmic barrier decomposition methods for semi-infinite programming. Technical Report 96–51, Faculty of Technical Mathematics and Informatics, Delft University of Technology, Delft, The Netherlands, 1996.

[18] Z.-Q. Luo. Analysis of a cutting plane method that uses weighted analytic center and multiple cuts. *SIAM Journal on Optimization*, 7:697–716, 1997.

[19] J. E. Mitchell. *Karmarkar's Algorithm and Combinatorial Optimization Problems*. PhD thesis, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY, 1988.

[20] J. E. Mitchell. An interior point column generation method for linear programming using shifted barriers. *SIAM Journal on Optimization*, 4:423–440, May 1994.

[21] J. E. Mitchell. Computational experience with an interior point cutting plane algorithm. Technical report, Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180–3590, February 1997. Revised: April 1997.

[22] J. E. Mitchell. Fixing variables and generating classical cutting planes when using an interior point branch and cut method to solve integer programming problems. *European Journal of Operational Research*, 97:139–148, 1997.

[23] J. E. Mitchell. An interior point cutting plane algorithm for Ising spin glass problems. Technical report, Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180–3590, July 1997.

[24] J. E. Mitchell and B. Borchers. Solving real-world linear ordering problems using a primal-dual interior point cutting plane method. *Annals of Operations Research*, 62:253–276, 1996.

[25] J. E. Mitchell and B. Borchers. Solving linear ordering problems with a combined interior point/simplex cutting plane algorithm. Technical report, Mathematical Sciences, Rensselaer Polytechnic Institute, Troy, NY 12180–3590, September 1997.

[26] J. E. Mitchell and S. Ramaswamy. An extension of Atkinson and Vaidya's algorithm that uses the central trajectory. Technical Report 37–93–387, DSES, Rensselaer Polytechnic Institute, Troy, NY 12180–3590, August 1993.

[27] J. E. Mitchell and M. J. Todd. Solving combinatorial optimization problems using Karmarkar's algorithm. *Mathematical Programming*, 56:245–284, 1992.

[28] S. Ramaswamy and J. E. Mitchell. On updating the analytic center after the addition of multiple cuts. Technical Report 37–94–423, DSES, Rensselaer Polytechnic Institute, Troy, NY 12180, October 1994.