Advanced linear programming

11.1, 11.2: Solving integer linear programming problems:

Totally Unimodular Matrices

Cutting planes

Marjan van den Akker

Universiteit Utrecht

[Faculty of Science
Information and Computing Sciences]

This lecture: More on solving ILPs

- Totally Unimodular Matrices: LP-relaxation is integral ☺

If LP-relaxation is fractional (usually)

- Cutting planes
- Branch-and cut

Totally Unimodular Matrices (TUM) (1)

- Basic feasible solution $(x_B, x_N) = (B^{-1}b, 0)$
- Assume that *B* and *b* are integral
- ***Observation:*** If the optimal basis of the LP-relaxation has *det(B)=-1 or +1,* then the linear programming relaxation solves the IP.
- ***Definition:*** A matrix is *totally unimodular* if every square submatrix has determinant +1, -1, or 0.
- ***Theorem:*** A matrix with in each row and column at most one +1 or -1 is TUM.

Totally Unimodular Matrices (TUM) (2)

- ***Theorem:*** A matrix with in each row and column at most one +1 or -1 is TUM.
- ***Theorem:*** A matrix A is TUM if and only if A^T *-A, (A,I),* and *(A,A)* are TUM

Totally Unimodular Matrices (TUM) (3)

- ***Theorem:*** A matrix *A* is TUM if
  1. $a_{ij} \in \{+1, 0, -1\}$ for all i, j
  2. Each column contains at most two nonzero coefficients
  3. There exists a partition (M_1, M_2) of the set M of rows such that each column j containing two nonzero coefficients satisfies

$$\sum_{i \in M_1} a_{ij} - \sum_{i \in M_2} a_{ij} = 0$$

Totally Unimodular Matrices (TUM) (4)

- ***Theorem:*** The linear program

$$\max\{cx \mid Ax \leq b, x \geq 0\}$$

has an integral optimal solution for all integer vectors *b* for which it has a finite optimal value if and only if *A* is TUM.

Totally Unimodular Matrices (TUM) (5)

- Assignment is TUM

Minimum cost network flow is TUM

- Directed graph (V,A)
- Unit flow cost c_{ij}
- Supply/demand b_i
  - supply positive, demand negative
- f_{ij}: flow on arc (i, j)

$$\min \sum_{(i,j)\in A} c_{ij} f_{ij}$$

s.t.

$$\sum_{k:(i,k)\in A} f_{ik} - \sum_{k:(k,i)\in A} f_{ik} = b_i$$

f_{ij} integral

Minimum cost network flow is TUM (2)

- Example b=(3,0,0,-2,4,-5)

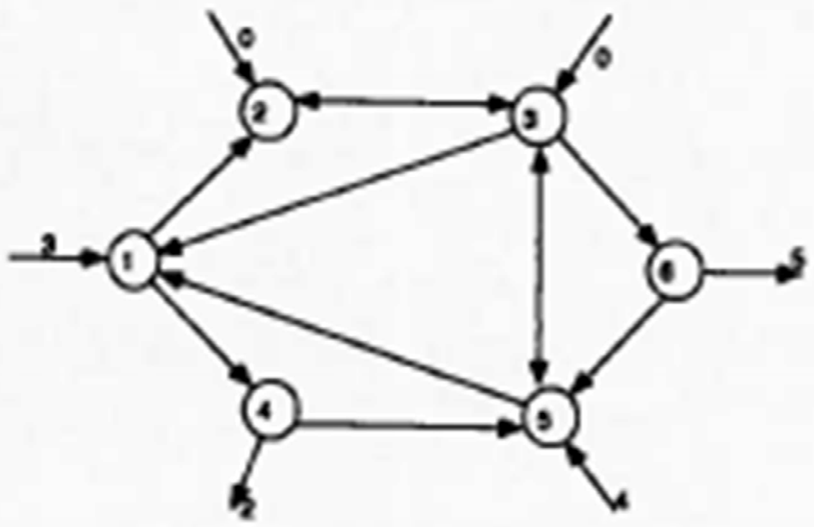


Fig. 3.1 Digraph for minimum cost network flow

Cutting plane algorithm

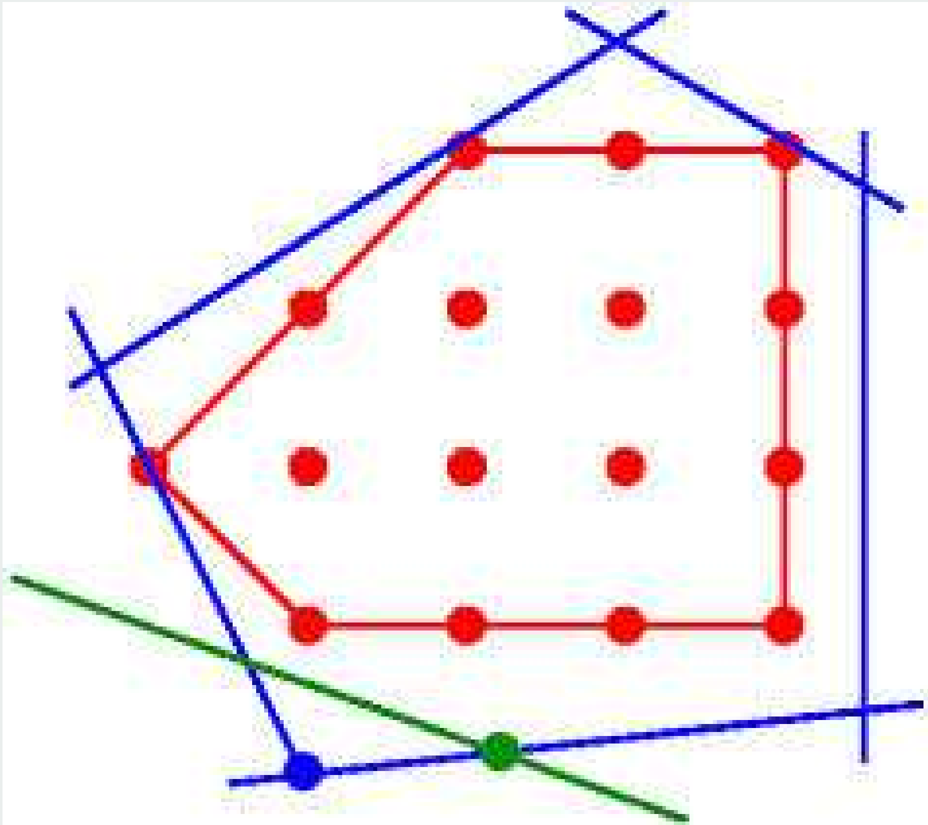
1. Solve the LP-relaxation. Let x^* be an optimal solution.
2. If x^* is integral, stop x^* optimal solution to the integer linear programming problem.
3. If not, add a *valid inequality* that is not satisfied by x^*

and go to Step 1. Solve separation problem.

Definition: *A valid inequality* is a linear constraint that is satisfied by all integral solutions.

If it cuts of fractional solutions it acts as *cutting plane*.

Cutting plane algorithm



Gomory cuts

- Take row from end tableau corresponding to fractional basic variable

$$x_i + \sum_{j \in N} \overline{a}_{ij} x_j = \overline{a}_{io} \text{ with } \overline{a}_{io} \text{ fractional}$$

- Gomory cut

$$x_i + \sum_{j \in N} \lfloor \overline{a}_{ij} \rfloor x_j \leq \lfloor \overline{a}_{io} \rfloor$$

- You can find the optimum by adding Gomory cuts, no practical method since you may need exponentially many steps

Universiteit Utrecht

[Faculty of **Science**
Information and Computing Sciences]

Problem-specific valid inequalities

- Usually classes of inequalities

- Cover inequalities for knapsack problems
- Weighted independent set
- Single-machine scheduling

“Cutting plane”

Branch-and-cut

- Combination of cutting planes and ILP solving by branch-and-bound
- Applied in well-known ILP-solvers:
  - CPLEX,
  - GUROBI,
  - EXPRES-MP
  - COIN-OR

Solving ILP by branch-and-bound or branch-and-cut

Let x* be the best known feasible solution

Search strategy

1. Select an active sub problem F_i (unevaluated node)
2. If F_i is infeasible: delete node
3. Compute upper bound $Z_{LP}(F_i)$ by solving LP-relaxation and feasible solution x_f (by rounding)

   If $Z_{LP}(F_i) \leq$ value x* delete node (bounding)

   If x_f is better than x*: update x*

   If solution x_{LP} to LP-relaxation is integral,

   then If x_{LP} is better than x*: update x* and node finished,

   otherwise split node into two new subproblems (branching)

4. Go to step 1

How many cuts? Which classes?

Primal heuristic

Branching strategy

Optional

This if for maximization problem, the book uses a minimization problem.

Branch-and-cut is a framework algorithm!!

Last century, tailoring to your own problem was necessary in most cases and a huge amount of research has been undertaken in doing this:

- Pre-processing
- Classes of valid inequalities
- How many cuts in each node?
- Search strategy
- Branching strategy
- Primal heuristics
- Reduced cost fixing

Now solvers like CPLEX and Gurobi successfully (and secretly) apply a lot of the above techniques.

- Tailoring of branch-and-cut sometimes successful, especially valid inequalities in case of a weak LP-relaxations from http://www.lnmb.nl/conferences/2015/programlnmbconference/LNMB-NGB_Bixby.pdf

1998 … A New Generation of MIP Codes

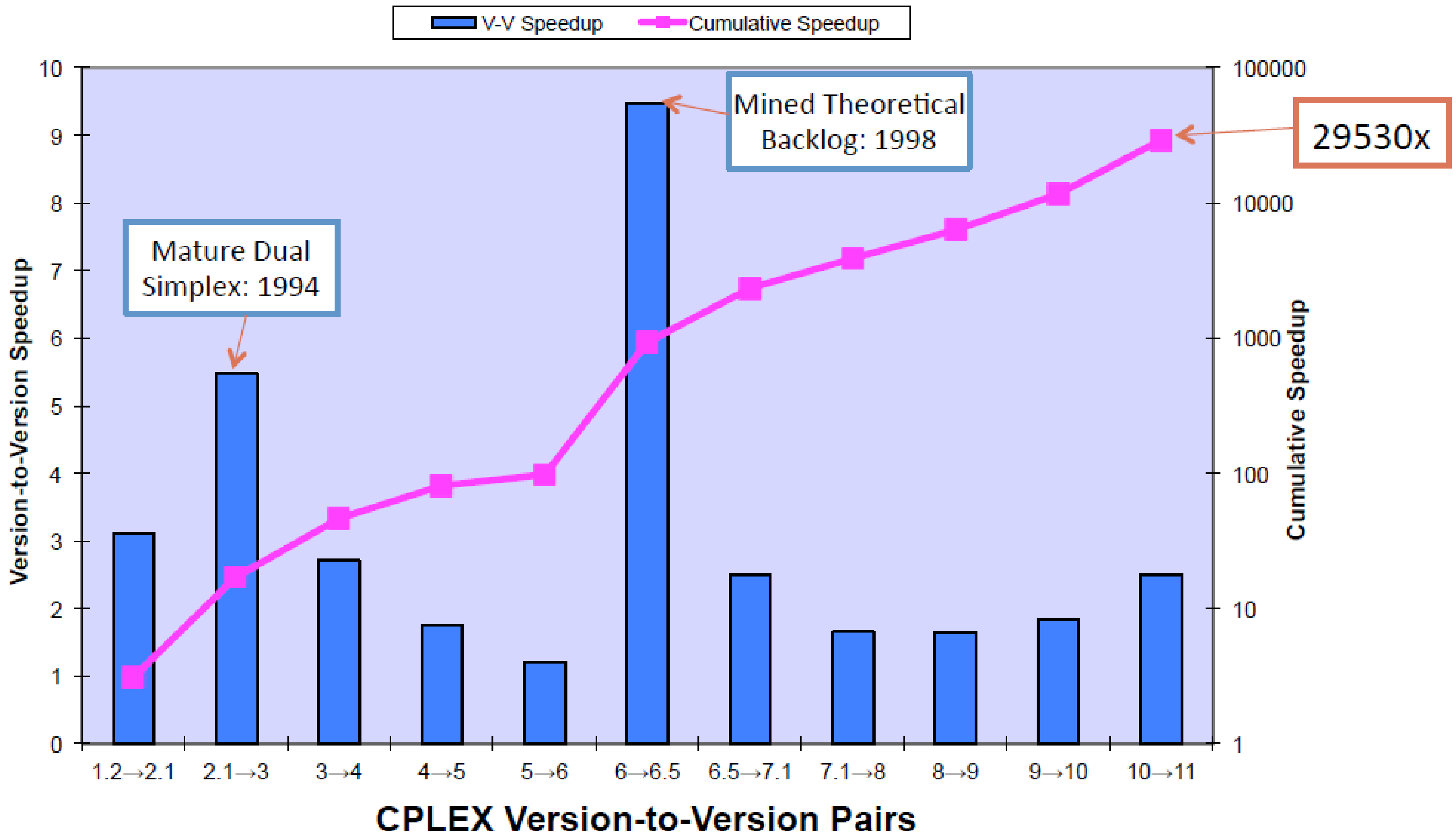
- Linear programming
  - Stable, robust dual simplex
- Variable/node selection
  - Influenced by traveling salesman problem
- Primal heuristics
  - 12 different tried at root
  - Retried based upon success
- Node presolve
  - Fast, incremental bound strengthening (very similar to Constraint Programming)
- Presolve – numerous small ideas
  - Probing in constraints:

    $\sum x_j \leq (\sum u_j)\, y, \;\; y = 0/1$

    $\rightarrow x_j \leq u_j y$ (for all j)
- Cutting planes
  - Gomory, mixed-integer rounding (MIR), knapsack covers, flow covers, cliques, GUB covers, implied bounds, zero-half cuts, path cuts

Gurobi Optimization

Speedups 1991-2008



10

World record exact TSP solving

- Branch-and-cut

CONCORDE:

http://www.math.uwaterloo.ca/tsp/concorde/index.html

Still we are talking about combinatorial optimization

- We have many, many, many variables
- Still large computation times
- Decomposition approaches often help
- **We now go to Chapter 6: Large-scale optimization**
  - *Decompositions for large LP's*
  - *Since these principles are usually used for solving ILP''s, Ch 10 was done first.*