Integral Column Generation

Issmail El Hallaoui (presenter), Adil Tahir, Guy Desaulniers

Polytechnique Montreal and GERAD
Department of Maths and Industrial Engineering

issmail.elhallaoui@polymtl.ca

May 24, 2016

Overview

Introduction

Consider the set partitioning problem ($\mathbb{P}$):

$$(\mathbb{P}) \quad min \quad \sum_{j \in N} c_j x_j \tag{1}$$

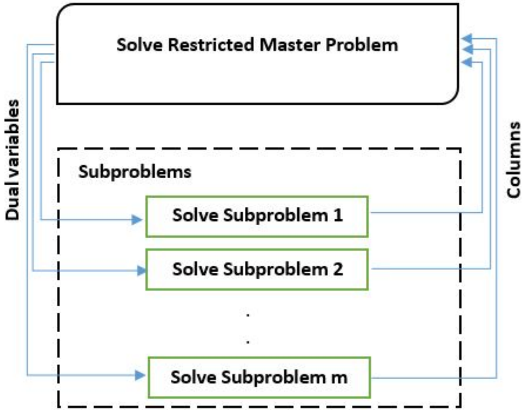
$$\text{s.t.:} \quad \sum_{j \in N} a_{ij} x_j = 1 \quad \forall i \in T \tag{2}$$

$$x_j \;\; binary \qquad \forall j \in N \tag{3}$$

1. T is a set of tasks (e.g. flights, bus trips, customers to visit), N is a set of columns (e.g. pilot/driver schedules), $A = (a_{i,j})$ is the constraint matrix with binary coefficients. **Let LP be the linear relaxation of $\mathbb{P}$.**
2. Many applications in the industry: vehicle and crew scheduling (air, rail, bus), data clustering, ...

Literature review: Branch-and-Price

- Branch-and-Price = Branch-and-Bound + Column Generation.



Many success stories since the late 80's: air, rail, bus industries...

Literature review: Branch-and-Price

- Branch-and-Price = Branch-and-Bound + Column Generation.

Advantages

1. Solve large programs(with special structure: part of the constraint matrix is diagonal by blocks).
2. Better lower bound.
3. Can handle nonlinearities in the subproblem.
4. Exact method.
5. ...

But, in the other hand ...

Literature review: Branch-and-Price

- Branch-and-Price = Branch-and-Bound + Column Generation.

Issues

1. Branch-and-price is a dual-fractional method (Letchford and Lodi (2002)).
2. The branching tree is huge in large reallife problems, integer solutions come too late!!!
3. Two many CG iterations because we consider at a given iteration only columns that are good for LP.
4. Plateau, yo-yo, tailing-off, heading-in, some headaches...

Some issues could be fixed ...

Literature review: Integral simplex

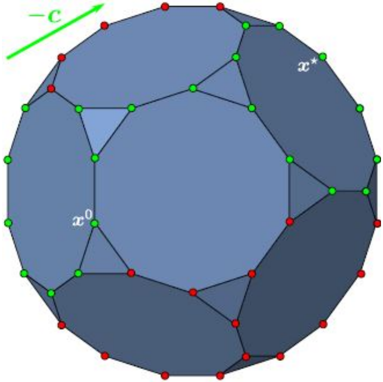


Figure : *LP* (Rosat et *al.*, 2016).

1. Trubin (1969): quasi-integrality; i.e. every edge of Conv($\mathbb{P}$) is an edge of *LP*.
2. Balas and Padberg (1972,1975): existence of an *integral* path with decreasing cost leading to an optimal integer solution.
3. Thompson (2002): Integral simplex method. Small instances solved : 163 constraints
4. Saxena (2003): generalization of Thompson(2002) results...

- The big issue: degeneracy.

Literature review: Integral simplex-and-Column generation

First integration with column generation

Rönnberg and Larsson (2009, 2014): all-integer column generation.

- All-integer pivots on columns and surrogate columns.
- Over generation.
- Optimality condition.
- Numerical results for generalized assignment problem (10 agents, 30 jobs).

- Still, the big issue is degeneracy.

Literature review: Degeneracy

Main points of view

1. **Avoid degeneracy**: increase the number of fractional variables; efficient against degeneracy but generally not good for branching.
2. **Profit from degeneracy**: to reduce the problem size and the number of fractional variables (basis reduced), by using for instance dynamic constraint/variable gggregation (Elhallaoui et *al.* (2005, 2010), Bouarab et *al.*(2014)).

In summary

- Solutions are either highly degenerate and "fractional" or nondegenerate and higly fractional!
- Let us get a look outside the box ...

Integral simplex using decomposition (ISUD): motivation

- Given a current integer solution $\bar{x}$, $S = \{j | \bar{x}_j \neq 0\}$.

Definition

A set $N' \subset N$ is **row-disjoint** if $A_{j_1} \cdot A_{j_2} = 0, \quad \forall (j_1, j_2) \in N' \times N', \quad j_1 \neq j_2$.

Definition

A column or a (positive) combination of columns is said to be **compatible** if it is linearly dependent on columns indexed in S. Otherwise, it is said to be **incompatible**.

Theorem

We move on to a next better adjacent integer solution **iff** *the combination of columns we pivot on is a negative reduced cost, compatible, minimal, and composed of row-disjoint columns.*

Integral simplex using decomposition (ISUD)

- We can now use this notion of compatibility to **decompose** the problem:

$A =$	A_S	A_{C_S}	A_{I_S}
		Compatible	Incompatible

Integral simplex using decomposition (ISUD)

- We can now use this notion of compatibility to **decompose** the problem:

$$A = \begin{array}{|c|c|c|} A_S & A_{C_S} & A_{I_S} \end{array}$$

Compatible | Incompatible

$\Downarrow$

RP

$$\min_{x_{C_S}} \quad c_{C_S} \cdot x_{C_S} \tag{4}$$

$$\text{sc:} \quad A_{C_S} x_{C_S} = e \tag{5}$$

$$x_{C_S} \in \{0,1\}^{|C_S|} \tag{6}$$

Integral simplex using decomposition (ISUD)

- We can now use this notion of compatibility to **decompose** the problem:

$A =$	A_S	A_{C_S}	A_{I_S}
		Compatible	Incompatible

$$\Downarrow$$

$$CP$$

reduced cost → $$\min_{v,\lambda} \sum_{j \in I_S} c_j v_j - \sum_{l \in S} c_l \lambda_l \quad (4)$$

compatibility constraints → st: $$\sum_{j \in I_S} v_j A_j - \sum_{l \in S} \lambda_l A_l = 0 \quad (5)$$

normalization constraint → $$\sum_{j \in I_S} w_j v_j = 1 \quad (6)$$

$$v \geq 0, \quad \lambda \in \mathbb{R}^{|S|} \quad (7)$$

Integral simplex using decomposition (ISUD): mechanics

Remark

- Positive (v_j) are entering variables, positive (λ_l) are leaving variables.
- The vector $d = ((v_j), -(\lambda_l), 0)$ is a descent direction if the reduced cost is negative.
- Constraint (5) could be replaced by $MAv = 0$. M is the **compatibility** matrix.
- So, CP becomes: min $\bar{c} \cdot v, MAv = 0, \sum_{j \in I_S} w_j v_j = 1, v \geq 0$.

Proposition

- The combination $v_j > 0$ found by CP is compatible and *minimal* (*non-decomposable*).
- If $\bar{x}$ is not optimal to $\mathbb{P}$, CP admits a row-disjoint minimal negative reduced cost combination as a solution.
- The new solution is "simply" obtained by exchanging the leaving variables (λ_l) with the entering ones (v_j).

Integral simplex using decomposition (ISUD): algorithm

ISUD algorithm

Step 0: Start with an initial integer solution.

Step 1: **Improve** the current integer solution with efficient pivots in the reduced problem.

Step 2: **Improve** the current integer solution with a compatible combination of columns (v_j) found by the complementary problem (making some branching if necessary).

Control: If Step 2 improves the solution, go to Step 1. Otherwise, the current solution is optimal.

Integral simplex using decomposition (ISUD): comments

Advantages

1. Primal exact approach.
2. CP finds frequently and rapidly row-disjoint solutions without suffering from degeneracy.
3. Faster than CPLEX on some hard instances.
4. Seed for other projects:
    - Primal cuts, normalization (Rosat et *al.* 2014, 2016);
    - Zooming approach, new formulation of CP (Zaghrouti et *al.* 2014, 2016);
    - Parallelization (Foutlane et al., under progress), and
    - **Integral column generation**.

Integral column generation: motivation

Some practical observations

1. We can obtain in practice initial starting green points with good primal information.
   - Follow vehicles (aircraft, bus...), i.e. derive a starting point (crew schedules) from vehicle routes (easy to find in general).
   - Use the perturbed plannig to derive a starting point in a reoptimization after perturbation.
   - Learn from historical data to get a starting point.
2. Local improvement of an integer solution is also highly desirable in practice. It is possible in practice to define area with potential improvement around the current green point by "accurately" measuring incompatibility.

Integral column generation: ISUD + Column generation

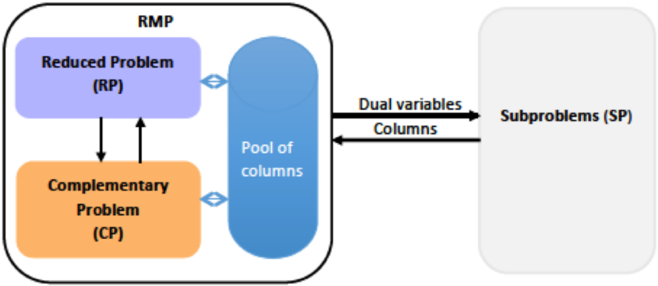


Figure : *ICG*[1]: Solving RMP by ISUD.

Integral column generation: theoretical insights

Proposition

There exists at least one vector of dual values $\alpha \in \mathbb{R}^m$ ("corresponding" to the current integer solution; could be derived from CP solution) such that all entering variables (in a descent direction) have a negative reduced cost.

Proposition

Let S^* be the index set of variables taking 1 in an optimal integer solution. There exists at least one dual vector ("corresponding" to the current integer solution) such that all variables indexed in $S^* \setminus S$ have a non-positive reduced cost.

Integral column generation: implementation

Some tips

- Generate **interesting**, from integrality point of view, columns as much as you can and handle them in a pool.
- Solve the restricted master problem with ISUD:
  - Use a multiphase strategy: consider at phase k columns j with $\|MA_j\|_1 \leq k$; increase k as needed.
- Loop until satisfaction.

Remark

- In phase k, the number of nonzero elements in any column of CP does not exceed $k+1$.

Integral column generation: numerical results

	Diving B&P				ICG^1			
Tasks	**Itr**	**Col**	**Obj**	**Time(s)**	**Itr**	**Col**	**Obj**	**Time(s)**
240	43	7744	60	0,6	3	22907	60	1
320	79	17614	477	2	4	24516	477	2
400	62	21069	60	4	3	101180	60	8
480	89	34748	477	7	5	78381	477	11
640	66	37236	477	14	7	173976	477	27
720	151	86648	747	45	5	137382	747	34
800	144	107707	447	47	4	260769	447	57

Table : ICG^1 vs. B&P (small instances)

Diving B&P is slightly better on these small instances. However, the number of iterations is drastically reduced!

Integral column generation: numerical results

	Diving B&P				ICG^1			
Tasks	**Itr**	**Cols**	**Obj**	**Time (s)**	**Itr**	**Cols**	**Obj**	**Time (s)**
960	294	150072	1378	115	5	143011	1378	42
1200	238	173895	173895	182	6	213042	1849	67
1200	249	317804	747	311	5	461902	747	159
1600	270	404115	880	689	6	568539	880	278
2000	359	705557	1259	3150	5	874990	1259	578

Table : ICG^1 vs. B&P (medium & large instances)

ICG is better on larger instances. Average reduction factors are:

- Time: 3
- Iterations: 85

References

Balas, E., Padberg, M. W. (1972). On the set-covering problem. Operations Research, 20(6), 1152-1161.

Balas, E., Padberg, M. (1975). On the set-covering problem: II. An algorithm for set partitioning. Operations Research, 23(1), 74-90.

Bouarab H., Elhallaoui, I., Metrane, A., Soumis, F. (2014). Dynamic constraint and variable aggregation in column generation. Cahiers du Gerad, G-2014-82, HEC Montral.

Elhallaoui, I., Metrane, A., Desaulniers, G., Soumis, F. (2011). An improved primal simplex algorithm for degenerate linear programs. INFORMS Journal on Computing, 23(4), 569-577.

Elhallaoui, I., Villeneuve, D., Soumis, F., Desaulniers, G. (2005). Dynamic aggregation of set-partitioning constraints in column generation. Operations Research, 53(4), 632-645.

References

- Elhallaoui, I., Metrane, A., Soumis, F., Desaulniers, G. (2010). Multi-phase dynamic constraint aggregation for set partitioning type problems. Mathematical Programming, 123(2), 345-370.
- Rönnberg, E., Larsson, T. (2009). Column generation in the integral simplex method. European Journal of Operational Research, 192(1), 333-342.
- Rönnberg, E., Larsson, T. (2014). All-integer column generation for set partitioning: Basic principles and extensions. European Journal of Operational Research, 233(3), 529-538.
- Rosat, S., Elhallaoui, I., Soumis, F., Chakour, D. (2016). Influence of the normalization constraint on the integral simplex using decomposition. Discrete Applied Mathematics.
- Rosat, S., Elhallaoui, I., Soumis, F., Lodi, A. (2014). Integral simplex using decomposition with primal cuts. In Symposium on Experimental Algorithms (pp. 22-33). Springer International Publishing.

References

Thompson, G. L. (2002). An integral simplex algorithm for solving combinatorial optimization problems. Computational optimization and applications, 22(3), 351-367.

Trubin, V. A. (1969). On a method of solution of integer linear programming problems of a special kind. In Soviet Mathematics Doklady, 10, 1544-1546.

Zaghrouti, A., Soumis, F., El Hallaoui, I. (2014). Integral simplex using decomposition for the set partitioning problem. Operations Research, 62(2), 435-449.

Zaghrouti, A., Soumis, F., El Hallaoui, I. (2014). Improving ILP Solutions by Zooming Around an Improving Direction. Cahiers du Gerad, G-2013-107, HEC Montral.

Conclusion

- Seems working good but needs some extensive testing and tuning.
- Could be generalized to SPP with side constraints. May be to general binary programs!
- Characterize fractional solutions better to penalize them in the normalization constraint!!

Thanks for your attention!