Accelerating the simplex algorithm via novel crash procedures

Nikolaos Ploskas, Nikolaos V. Sahinidis, Nikolaos Samaras

Dantzig's simplex algorithm was initially proposed in the late 1940s [1]. Many variants of this algorithm have been studied extensively by numerous researchers and have become core routines in many optimization algorithms for linear, integer and nonlinear programming [2, 3]. Modern linear programming solvers such as CPLEX [4] are currently capable of routinely solving linear programs with millions of constraints and variables using the simplex algorithm. Most Fortune 500 companies are using the simplex algorithm in their decision tools. The simplex algorithm was recently included in the list of the top-10 algorithms of the twentieth century [5]. Given this level of activity in the field and presumed maturity of the simplex algorithm, one would think that it would be very difficult, if at all possible, to improve significantly current implementations of the simplex algorithm. In this paper, we show that a novel initialization strategy provides a means of considerable speed up of the simplex algorithm.

Most crash procedures use triangulation and sparsification concepts [6, 7]. Taking into account that the initial basis will be factorized using LU decomposition, most crash procedures form a nearly-triangular and sparse basis that is likely to limit the number of subsequent fill-ins. In this paper, we propose six algorithms for constructing an initial basis. The algorithms rely on triangulation and fill-reducing ordering techniques that are applied prior to LU factorization. We compare the performance of the CPLEX primal and dual simplex algorithms using the proposed starting bases against CPLEX using its default crash procedure over a set of 62 large benchmarks. The best proposed algorithm produces remarkably sparse starting bases, and results in 7% and 6% average reduction of the execution time of CPLEX's primal and dual simplex algorithm, respectively. We also present results for very large and degenerate linear programming problems for which our best algorithm leads CPLEX's primal and dual simplex algorithm to perform an order of magnitude faster than the CPLEX default crash procedures.

References cited:

[1] Dantzig, G. B. (1963). Linear programming and extensions. Princeton, NJ: Princeton University Press.

[2] Bixby, R. E. (2002). Solving real-world linear programs: A decade and more of progress. Operations research, 50, 3–15.

[3] Bixby, R. E. (2012). A brief history of linear and mixed-integer programming computation. Documenta Mathematica, 107–121.

[4] IBM (2018). IBM ILOG CPLEX V12.6.1. Available online: https://www.ibm.com/support/knowledgecenter/SSSA5P_12.6.1/ilog.odms.cplex.help/CPLEX/homepages/CPLEX.html

[5] Dongarra, J., Sullivan, F. (2000). Guest editors' introduction: The top 10 algorithms. Computing in Science & Engineering, 2(1), 22–23.

[6] Bixby, R. E. (1992). Implementing the simplex method: The initial basis. ORSA Journal on Computing, 4, 267–284.

[7] Gould, N. I. M., Reid, J. K. (1989). New crash procedures for large systems of linear constraints. Mathematical Programming, 45, 475–501.