Solving Linear Problems Using Simplex Method

[1]Usha K Patil, [2] Rashmi M
[1][2] Assistant Professor Department of Computer Science and Engineering,
GSSS Institute of Engineering and Technology for Women, Mysuru
[1]Patilusha507@gmail.com, [2] rashmigssscampus@gmail.com

***Abstract* - The main aim of our application is to implement Simplex method using C. The Simplex Method is the classic method for solving LP problems, one of the most important algorithms ever invented. Invented by George Danzig in 1947 (Stanford University).Based on the iterative improvement idea. Generates a sequence of adjacent points of the problem's feasible region with improving values of the objective function until no further improvement is possible. This paper presents the result of Simplex method using C project. The objective and scope of this project is to manage and simplify the task and reduce the paper work.**

***Keywords*—Simplex method ;optimization; Constraints; objective function; variables;hybrid computing; GPU computing; parallel; slack variable; surplus variables;**

I. INTRODUCTION

The Simplex method is considered to be one of the most significant algorithms of the last century. It is a method for solving the linear optimization problem and its worst case complexity is exponential in the number of variables However, it is very efficient in practice and converges in polynomial time for many input problems, including certain classes of randomly generated problems. Apart from the basic Simplex method for the optimization problem, there are many other variants, including a decision variant that decides if a set of linear constraints is satisfiable or not.

The Simplex method has a wide range of applications, in direct sorts of optimization problems, but also in software and hardware verification.
In this paper, we describe how a decision version of the Simplex method can be used in automated detection of buffer overflows in programming language C.

This work was partially supported by Serbian Ministry of Science grant 144030.For instance, the journal Computing in Science and Engineering listed it as one of the top 10 algorithms of the century, data in a data storage area than it was intended to hold.

The Simplex method is the most common way to solve large LP problems. *Simplex* is a mathematical term. In one dimension, a Simplex is a line segment connecting two points. In two dimensions, a Simplex is a triangle formed by joining the points. A three dimensional Simplex is a four-sided pyramid having four corners

Maximization Method The underlying concepts are geometrical, but the solution algorithm, developed by George Danzig in 1947, is an algebraic procedure. As with the graphical finds the most attractive corner of the feasible region to solve the LP problem. Remember, any LP problem having a solution must have an optimal solution that corresponds to a corner, although there may be multiple or alternative optimal solutions. Simplex usually starts at the corner that represents doing nothing.

It moves to the neighboring corner that best improves the solution. It does this over and over again, making the greatest possible improvement each time. When no more improvements can be made, the most attractive corner corresponding to the
optimal solution has been found.

Minimization Method A moderately sized LP with 10 products and 10 resource constraints would involve nearly 200,000 corners. An LP problem 10 times this size would have more than a trillion corners. Fortunately, the search procedure for the Simplex method is efficient
enough that only about 20 of the 200,000 corners are searched to find the optimal solution.

In the real world, computer software is used to solve LP problems using the Simplex method, but you will better understand the results if you understand how the Simplex method work.

II. LITRETURE SURVEY

This section presents the analysis of different Existing system.

- G. Yarmish, “The Simplex method applied to wavelet decomposition,”in Proc. of the International Conference on Applied Mathematics, Dallas, USA, 226–228, November 2006.

- The algorithm described in Section 3 has been experimentally implemented. In this Section, the numerical experiments are presented. It must be mentioned that the computational results demonstrate a speedup for traditional Simplex algorithm on dense linear programs.

- All test runs were carried out on 16 uniprocessors Intel Pentium III 500MHz with 512 KB L2 Cache. The processors were interconnected using Fast Ethernet and Scalable Coherent Interface (SCI). Furthermore, the machine precision was 32 decimal digits. The reported CPU times were measured in seconds. MPI implementation MPICH v.1.2.6 was used and appropriately configured for our cluster. Usage of this machine was provided by the National Technical University of Athens, School of Electrical and Computer Engineering.

- J. Eckstein, I. Bodurglu, L. Polymenakos, and D. Goldfarb,“Data-Parallel Implementations of Dense Simplex Methods on the Connection Machine CM-2,” ORSA Journal on Computing,vol. 7,4:434–449, 2010.

- In this article we have proposed a parallel implementation of the Simplex method for solving linear programming problems on CPU-GPU system with CUDA. The parallel implementation has been performed by optimizing the different steps of the Simplex algorithm. Computational results show that our implementation in double precision on CPU-GPU system is efficient since for large non-sparse linear programming problems, we have obtained stable speedups around 12.5.

- Our approach permits one also to solve problems of size 8000 _ 8000 without exceeding the memory capacity of the GPU. In future work, we plan to test larger LP problems on multi GPU architecture. We plan also to implement the revised Simplex algorithm on GPU without using CUBLAS or LAPACK libraries in order to go on further in the optimization of the parallel implementation.

- R.S.Garfinkel,D.L.Nemhauser,IntegerProgramming,Wiley-Interscience, 1972.

Finding the entering or leaving variables results in finding a minimum within a set of values. This can be done on GPU via reduction techniques. However, our experiments showed that we obtain better performance by doing this step sequentially on the CPU. Indeed, the size of the tested problems and the double precision operations lead to worst efficiency for the parallel approach.

More explicitly, finding a minimum in a row of 10000 values, which corresponds to the maximal row size treated in our experiments, takes an average time of 0:27ms on CPU and 3:17ms on GPU at each step of the Simplex algorithm. Furthermore, the use of atomic functions of CUDA is not possible in this case since they do not support double precision operations.

Thus, this step is implemented in CPU and the minimum index is thereafter communicated to the GPU. In step 1 of the Simplex algorithm, the first line of the Simplex tableau is simply communicated to the CPU. However, the step 2 requires the processing of a column of ratios. This is done in parallel by Kernel 1 and the ratio column _ is communicated to the CPU. Since step 3 requires the column k, the column of entering variable of the Simplex tableau, Kernel 1 is also used to get the “old” Columnk and to store it in the device memory in order to avoid the case of memory conflict.

III. EXISTING SYSTEM AND PROPOSED SYSTEM

- Manually done using ink and paper.
- Consumes more effort and time.
- Difficult to keep track of a particular problem.
- Difficult to maintain all data.
- Since it is done manually, it is costly

- The Proposed System architecture
- Reduces time consumption
- Easy to handle and feasible
- Easy to operate
- Fast and convenient
- Cost reduction

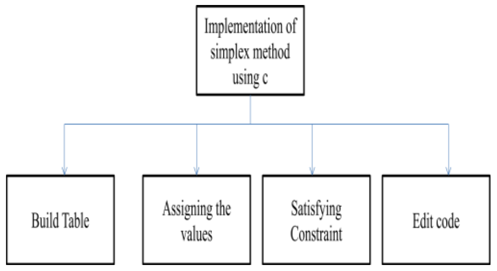


Fig1: Architecture of Simplex method

The above figure shows the architecture of the proposed system. It contains four modules.

Build table, assigning the values, satisfying constraint, edit code.

IV ALGORITHM

The key solution concepts

- ❖ Step 1: the Simplex method focuses on CPF solutions.
- ❖ Step 2: the Simplex method is an iterative algorithm (a systematic solution procedure that keeps repeating a fixed series of steps, called, an iteration, until a desired result has been obtained) with the following structure:
- ❖ Step 3:initialization set up to start iterations including solutions finding an initial CPF
- ❖ Step 4:optimality test,is the current CPF solution if no,performs iteration else stop.
- ❖ Step 5:perform an iteration to find better CPF solution .

V IMPLEMENTATION

Implementation is the realization of an application, or execution of a plan, idea, model, design, specification, standard, algorithm, or policy. In computer science, an implementation is a realization of a technical specification or algorithm as a program, software component, or other computer system through computer programming and deployment.

Many implementations may exist for a given specification or standard. For example, web browser contains implementations of World Wide Web consortium-recommended specifications, and software development tools contain implementations of programming languages. The implementation stage requires the following tasks.

- ❖ Careful Planning.
- ❖ Investigation of system and constraints.
- ❖ Design of methods to achieve the changeover.
- ❖ Evaluation of the changeover method.
- ❖ Correct decisions regarding selection of the platform.

```
void Simplex(Table *tab) {
  int loop=0;
  add_slack_variables(tab);
  check_b_positive(tab);
  print_table(tab,"Padded with slack variables");
  while( ++loop ) {
    int pivot_col, pivot_row;

    pivot_col = find_pivot_column(tab);
    if( pivot_col < 0 ) {
      printf("Found optimal value=A[0,0]=%3.2lf (no negatives in row 0).\n",
        tab->mat[0][0]);
      print_optimal_vector(tab, "Optimal vector");
      break;
    }
    printf("Entering variable x%d to be made basic, so pivot_col=%d.\n",
      pivot_col, pivot_col);

    pivot_row = find_pivot_row(tab, pivot_col);
    if (pivot_row < 0) {
      printf("unbounded (no pivot_row).\n");
      break;
    }
    printf("Leaving variable x%d, so pivot_row=%d\n", pivot_row, pivot_row);

    pivot_on(tab, pivot_row, pivot_col);
    print_table(tab,"After pivoting");
    print_optimal_vector(tab, "Basic feasible solution");

    if(loop > 20) {
      printf("Too many iterations > %d.\n", loop);
      break;
    }
  }
}
```

VI DESIGN

High level design

Design process is nothing the representation of the system, or a process of a producing a model, which will be used to developed or build the system. The input for the design process is the SRS and the output is “Design of the proposed system”. While SRS is entirely in problem domain, design is the first step in moving from the problem domain to solution domain. Design is essentially a bridge between the requirement specification and the final solution for the satisfying the requirements. Thus it is essentially a blue print of a solution for the system.

High Level Design (HLD) is the overall system design- covering the system architecture and database design. It describes the relation between various modules and functions of the system. Data flow, flow charts and data structures are covered under HLD.

- ❖ A flowchart is a type of diagram that represents an algorithm, workflow or process

Low Level Design

Low Level Design (LLD) defines the actual logic for each and every component of the system. Class diagrams with all the methods and relation between classes comes under LLD. LLD describes each and every module in an elaborate manner so that the programmer can directly code the program based on this.

There will be at least one document for each module. The LLD will contain detailed functional logic of the module including their type and size all interface details with complete API references (both requests and responses) - all dependency issues error message listings complete input and outputs for a module.

Pivot row:
Initialization:

- pivot_col, pivot_row;

Maximization:
Computes the leaving and unbounded variables.
If(pivot<0)
Enters the loop ;
Else
Terminates;

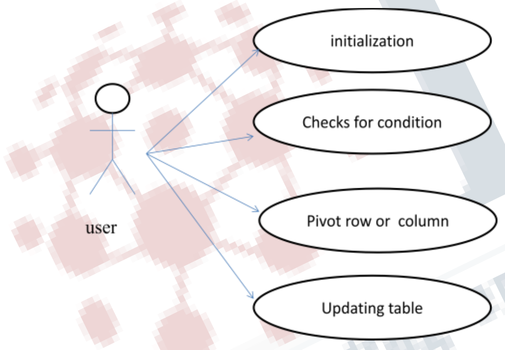
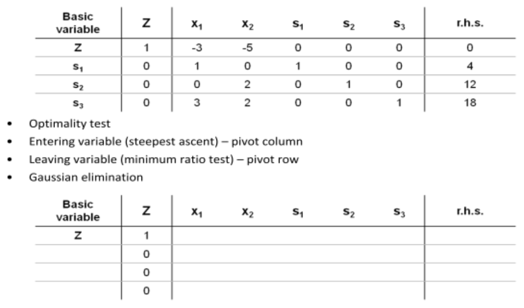


Fig 2: Use case diagram of Simplex method system

VII SAMPLE TEST CASES

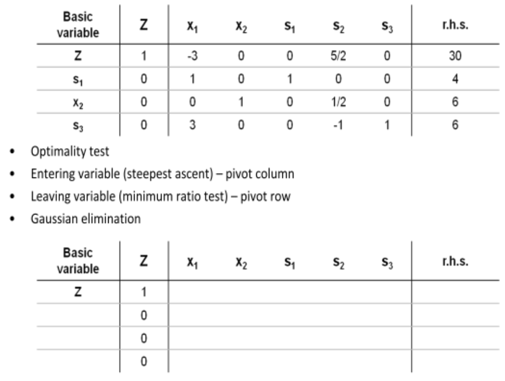


Basic variable	Z	x_1	x_2	s_1	s_2	s_3	r.h.s.
Z	1	-3	-5	0	0	0	0
s_1	0	1	0	1	0	0	4
s_2	0	0	2	0	1	0	12
s_3	0	3	2	0	0	1	18

- Optimality test
- Entering variable (steepest ascent) – pivot column
- Leaving variable (minimum ratio test) – pivot row
- Gaussian elimination

Basic variable	Z	x_1	x_2	s_1	s_2	s_3	r.h.s.
Z	1						
	0						
	0						
	0						

Fig 3 sample test cases

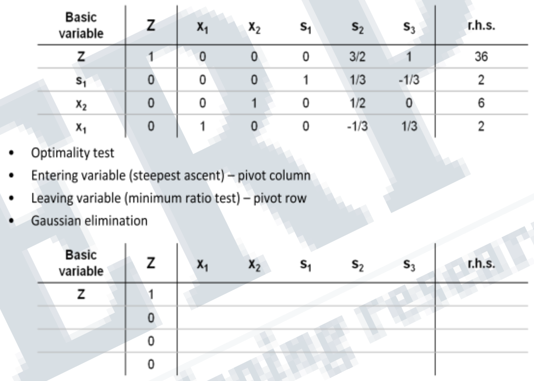


Basic variable	Z	x_1	x_2	s_1	s_2	s_3	r.h.s.
Z	1	-3	0	0	5/2	0	30
s_1	0	1	0	1	0	0	4
x_2	0	0	1	0	1/2	0	6
s_3	0	3	0	0	-1	1	6

- Optimality test
- Entering variable (steepest ascent) – pivot column
- Leaving variable (minimum ratio test) – pivot row
- Gaussian elimination

Basic variable	Z	x_1	x_2	s_1	s_2	s_3	r.h.s.
Z	1						
	0						
	0						
	0						

Fig 4 sample test cases



Basic variable	Z	x_1	x_2	s_1	s_2	s_3	r.h.s.
Z	1	0	0	0	3/2	1	36
s_1	0	0	0	1	1/3	-1/3	2
x_2	0	0	1	0	1/2	0	6
x_1	0	1	0	0	-1/3	1/3	2

- Optimality test
- Entering variable (steepest ascent) – pivot column
- Leaving variable (minimum ratio test) – pivot row
- Gaussian elimination

Basic variable	Z	x_1	x_2	s_1	s_2	s_3	r.h.s.
Z	1						
	0						
	0						
	0						

Fig 5 sample test cases

VIII CONCLUSION

The Simplex algorithm is a well known method to solve linear programming (LP) problems. In this paper, we propose a parallel implementation of the Simplex on CPU-GPU systems via CUDA. Double precision implementation is used in order to improve the quality solutions. Computational tests have been carried out on randomly generated instances for non-sparse LP problems.

Acknowledgement

This work is a part of computer engineering projects at the Department of Computer Engineering, GSSS Institute of Engineering and Technology for Women. Authors would like to thank guide who helped us to evaluate the proposed system.

REFERENCES

[1] V. Boyer, D. El Baz, M. Elkihel, “Dense dynamic programming on multi GPU,” in Proc. of the 19th International Conference on Parallel Distributed and

networked-based Processing, PDP 2011, Ayia Napa, Cyprus, 545–551, February 2011.

[2] E. B. Ford, “Parallel algorithm for solving Keplers equation on graphics processing units: application to analysis of Doppler exoplanet searches,” New Astronomy, 14:406-412, 2009.

[3] G. B. Dantzig, *Linear Programming and Extensions*. NJ: Princeton, Princeton University Press, 1963.

[4] K. H. Borgwardt, “Some distribution independent results about the asymptotic order of the average number of pivot steps in the Simplex
method”, *Mathematics of Operations Research*, vol. 7, no. 3, pp. 441-462, 198

[5] A. Schrijver, Theory of linear and integer programming. John Wiley& Sons, 1986.

[6] S. Lin and D. Costello, Jr., Error Control Coding, 2nd ed. Upper Saddle sRiver, NJ: Prentice-Hall, Inc., 2004.

[7] D. G. Spampinato, A. C. Elster, “Linear optimization on modern
GPUs,” in Proc. of the 23rd IEEE International Parallel
and Distributed Processing Symposium, (IPDPS 2009), Rome,
Italy, May 2009.

[8] CUDA - CUBLAS Library 2.0, NVIDIA Corporation,

[9] LAPACK Library, http://www.culatools.com/