"OPTIMIZATION SOLVERS ON PLC; FEASIBILITY STUDY"

A Major Project Report

Submitted in Partial Fulfillment of the Requirements for the degree

Of

BACHELOR OF TECHNOLOGY
IN
INSTRUMENTATION AND CONTROL ENGINEERING

By
Jyot Buch (10BIC047)

Under the Guidance of
Dr. JayeshBarve

DEPARTMENT OF ELECTRICAL ENGINEERING
INSTITUTE OF TECHNOLOGY
NIRMA UNIVERSITY
Ahmedabad 382 481

May 2014

CERTIFICATE

THIS IS TO CERTIFY THAT THE PROJECT REPORT ENTITLED **"ADVANCED OPTIMIZATION ON PLC"** SUBMITTED BY **MR. JYOT BUCH (10BIC047)** TOWARDS THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE IN **BACHELOR OF TECHNOLOGY (INSTRUMENTATION & CONTROL ENGINEERING)** OF **NIRMA UNIVERSITY OF SCIENCE AND TECHNOLOGY** IS THE RECORD OF WORK CARRIED OUT BY HIM UNDER MY/OUR SUPERVISION AND GUIDANCE. THE WORK SUBMITTED HAS IN OUR OPINION REACHED A LEVEL REQUIRED FOR BEING ACCEPTED FOR EXAMINATION. THE RESULTS EMBODIED IN THIS PROJECT WORK TO THE BEST OF MY/OUR KNOWLEDGE HAVE NOT BEEN SUBMITTED TO ANY OTHER UNIVERSITY OR INSTITUTION FOR AWARD OF ANY DEGREE OR DIPLOMA.

DR. AMIT PUROHIT **ABB GISL**	**DR. JAYESHBARVE** **PROJECT GUIDE**	**DR. DIPAK M ADHYARU** **SECTION HEAD (IC)**	**DR. P N TEKWANI** **HOD (EE)**

DATE:

Acknowledgements

I would like to express my sincere thanks to ABB Corporate Research Center, Bangalore for providing me the opportunity and resources to carry out the project work. I am deeply indebted to Mr. Amit Purohit for his immense help, cooperation and invaluable suggestions and for being a constant source of motivation and support.

I am very thankful to department of Instrumentation & Control of Institute of Technology under Nirma University for allowing me to explore the research opportunity at ABB. I would like to thank my internal supervisor Dr. Jayesh Barve for his guidance and continuous encouragement during this Research.

I would like to thank Dr. Hans Joachim Ferreau and Dr. Jan Poland from ABB Corporate Research Center, Switzerland for their valuable support and guidance in Optimization domain. I am also thankful to Mr. Vinay Kariwala for his farsighted suggestions and guidance. I am also grateful to him for explaining the vision and scope of the project. I would also like to thank Shalini Shekar for her support as HR during this internship.

Moreover, I very much appreciated the opportunity to work on optimization solvers that are open-source. No software is contribution of an individual. Thus, I also would like to thank everybody who supported and contributed to the solvers packages which are used in this thesis.

Last but not the least, I would like to say a word of thanks to all those people be it the ABB scientists, engineers, other employees, support staff and the co-interns who have guided me through this project.

My apologies if I have inadvertently omitted anyone to whom acknowledgement is due.

Jyot Buch
May 2014

Abstract

Natural & Economical resources are limited, optimum use of them is adequate. Optimization in advanced control deals with implementation of algorithms on Process Control Computers in order to increase the efficiency of a plant for the optimum use of resources in terms of energy consumption, cost effectiveness, environmental impact, performance criteria etc.

But this approach of implementation becomes really challengeable when plant is located at harsh areas where no computers for advanced control & optimization available. As these areas are more prone to have unpredicted disturbances of humidity, dust, electromagnetic interferences and vibrations, PCs are not available in such areas, also the response of PC based system is slow as it has to do many tasks. This motivated us to implement optimization algorithm on the PLC. Compared to PC, PLC as a standalone optimizer can do this task more efficiently and effectively. Also, the latency introduced by the communication between PLC & PC in conventional approach can be removed if optimization algorithm can run on PLC. Only the required data can be uploaded on the PC or server.

During the Research, we found that so far the control dominated work is implemented on PLC like Model predictive control, PID control, Cascade control, Feed forward/feedback control, etc. So, in this thesis optimization algorithms are implemented on ABB AC-500 series PLC and created as an inbuilt library. The approach of embedded optimization is discussed.

Keywords: PLC, Embedded optimization, Library development, Optimization on PLC, Applications of PLC based optimization,

Abbreviations

PLC	*Programmable Logic Controller-* used as a general name for industrial control systems
APC	*Advanced Process control*
IEC	*International Electro technical Commission* – an international standards organization dealing with electrical, electronic and related technologies.
IEC 61131	An open international standard for PLCs, known as IEC 1131 before the change in the numbering system by IEC
IEC 61131-3	Part 3 of the IEC 61131 defines the PLC programming language standards
PRNG	Pseudo Random Number Generator
MPC	Model Predictive Control
CBP	Control Builder Plus
PID	Proportional Integral & Derivative
GISL	Global Industries and Services Ltd.
INCRC	India Corporate Research Center
C&O	Control & Optimization
ISS	Industrial Software Solutions
IC	Industrial Communications
IDC	India Development Center
PA	Process Automation

MP	Measurement Products
IS	Instrumentation Systems
INOPC	India Operation Center
DMC	Discrete Motion Controllers
MILP	Mixed Integer Linear Programming

List of Figures

Contents

Chapter 1

Introduction to Organization

ABB is a Multinational corporation headquartered in Zurich, Switzerland; it is a leader in Power & automation technologies that enable utility and industry customers to improve performance while lowering the environment impact. ABB is one of the largest conglomerates in the world.

ABB is the world's largest builder of electricity grids and is active in many sectors, its core businesses being in power and automation technologies. ABB is one of the few large companies that have successfully implemented the matrix structure in their organization. The company has one corporate division and five production divisions since reorganization in January 2010.

1) Discrete Automation and Motion

The division Discrete Automation and Motion provides products and services for industrial production. It includes electric motors, generators, drives, programmable logic controllers (PLCs), wind generator, solar power inverter, UPS products, power electronics and industrial robots.

2) Low Voltage Products

This division manufactures low-voltage circuit breakers, switches, and control products, wiring accessories, enclosures and cable systems to protect people, installations and electronic equipment from electrical overload. The division further makes KNX systems that integrate and automate a building's electrical installations, ventilation systems, and security and data communication networks. Low Voltage Products also incorporates a Low Voltage Systems unit manufacturing low voltage switchgear and motor control centers.

3) Power Products

Power products are the key components for the transmission and distribution of electricity. The division incorporates ABB's manufacturing network for transformers, switchgear, circuit breakers, cables, and associated high voltage and medium voltage equipment such as digital protective relays. It also offers maintenance services. The division is subdivided into three business units - High Voltage Products, Medium Voltage Products and Transformers.

4) Power Systems

Power Systems offers turnkey systems and service for power transmission and distribution grids, and for power plants. Electrical substations and substation automation systems are key areas. Additional highlights include flexible AC transmission systems (FACTS), high-voltage direct current (HVDC) systems and network management systems. In power generation, Power Systems offers the instrumentation, control and electrification of power plants. The division is subdivided into four business units - Grid Systems, Substations, Network Management, and Power Generation.

5) Process Automation

The main focus of this ABB business is to provide customers with systems for control, plant optimization, and industry-specific automation applications. The industries served include oil and gas, power, chemicals and pharmaceuticals, pulp and paper, metals and minerals, marine and turbo charging. By 2014, PA will consist of six business units: Turbo charging, Marine, Control Technology (the world's No 1DCS supplier) Measurement Products & Services, and Industrial Solutions (consolidation of O&G, Mining, and PMC into one business unit).

Global Research Lab

By linking and integrating operations at the corporate research centers located at different locations, the Global Research lab brings together an international team of highly skilled scientists in an innovative climate.

The research lab consists of nine research areas, each of which is led by a global manager.

- Communication
- Power Electronics
- Sensors
- Control
- Electromagnetism
- Materials
- Software
- Switching
- Mechanics

Evolutionary Innovation at ABB deals with incremental improvement of existing product whereas *Disruptive innovation* allows ABB to pursue Technology & Business in areas that ABB is not active today.

Corporate Research Centers

Each of the seven research center across the globe, scientist are working hard toward novel technological achievements that will help strengthen the five ABB Group divisions.

→China- (Beijing & Shanghai)
→Germany- (Ladenburg)
→India- (Bangalore)
→Poland- (Krakow)
→Sweden- (Vasteras)
→Switzerland- (Dattwil near Zurich)
→United States- (North Carolina)

ABB Group

ABB in India is headquartered at Bangalore, Karnataka. The corporate Research center is also located at Bangalore, Whitefield, which is known as ABB GISL.

Research at ABB, is the key aspect of their pioneering power and Automation Technologies. It contains mainly 3 groups.

The group of Control & Optimization (C/O) formed by 18 scientists, closely deals with other Research centers worldwide and contributes innovations for ABB's Control system Community.

-See Abbreviations for details of other divisions

Chapter-2

Project overview

Objective:

The objective of this research is to evaluate the performance of PLC based optimization and to compare it with traditional PC based optimization. Moreover, to enhance the capability of higher end PLC if advanced control algorithm can be run on it. There are potential benefits if embedded optimization can be run on PLC platform as it is well adopted as an industrial controller.

Research Questions:

I. Why PLC for Advanced Optimization?

II. Using IEC 61131-3 PLC standards it is difficult to program for optimization algorithms, then what are the other way to program PLC for advanced algorithms?

III. What are the challenges while implementing APC algorithms?

IV. What are the other advantages of PLC based optimization?

V. Which type of applications, it can be put in to practice?

Outline of Report:

This introduction is followed by summarizing previous work for PLC implementation of advanced control algorithms and ending up with some motivation for this project. In Chapter 3 a generic approach for implementation of Optimization algorithms is discussed with required justifications. In Chapter 4 the practical scope and target industries are discussed. Chapter 5 concludes the thesis and discuss about future scope in this domain.

Previous Work

As stated in [19] online MPC is implemented, in which at each step, an optimization problem is solved, on both a programmable automation controller PAC (NI compact RIO) and a programmable logic controller PLC (S7-319, Siemens). Three different optimization routines to solve the quadratic program were investigated with respect to their applicability on these devices. An “air heating” setup was built and selected as a small-scale multi-input single-output system. It turns out that the code generator CVXGEN is not suited for the PLC as the required programming language is not available and the programming concept with pre-allocated memory consumes too much memory. The Hildreth and qpOASES algorithms successfully controlled the setup running on the PLC hardware.

Reference [20] deals with real-time implementation of Model Predictive Control (MPC) of a fan heater system using Programmable Logic Controller (PLC) platform. The MPC problem is solved using parametric programming techniques, which encode the optimal control moves as a lookup table. The challenge then becomes how to implement such a table on a memory-restricted device. SIMATIC S7-200 micro PLC is used for implementation purpose and an air heating arrangement is modeled for a case study.

As optimization problems we can consider: to minimize or maximize a specific parameter and thus result to an equation to be used for bits and power assignment on the OFDM (Orthogonal frequency division multiplexing) carriers described in [21]. There are three categories of practical interest. In this paper one algorithm of each category was chosen and compared to the others in a real PLC environment. There is always a trade off in communications speed and the error at the receiver side, this paper tries to optimize the communication and compare H-H algorithm (Piazzo's algorithm), L-C algorithm, and F-H algorithm.

Research thesis [22] broadly emphasis on Time synchronized simulations and virtual manufacturing concept. It also summarize that HIL (Hardware in a loop) simulation with real PLC is an excellent method of obtaining off-line parameter tuning of complex control functions. By connecting two PLCs together we can achieve the validation, i.e. the first PLC acts as a model of the manufacturing system, whilst the second executes the control functions. This is possible for small plants only because big process plants are difficult to implement on PLC. Three global, direct search optimization algorithms which can handle many parameters and converge in few evaluations is suggested i.e. factorial design, direct algorithm, Nelder-Mead simplex algorithm. Finally using a steel metal press line they have explained the virtual manufacturing.

In Literature [23], optimized fuzzy controller is presented for the control of the environmental parameters at the building zone level. The occupants' preferences are monitored via a smart card unit. Genetic algorithm optimization techniques are applied to shift properly the membership functions of the fuzzy controller in order to satisfy the occupants' preferences while minimizing energy consumption. The implementation of the system integrates a smart card unit, sensors, actuators, interfaces, a programmable logic controller (PLC) (AMBER S.A), local operating network (LON) modules and devices, and a central PC which monitors the performance of the system.

In [24] Model Predictive Control (MPC) on a Programmable Automation Controller (PAC) The Compact Rio is programmed with Lab-VIEW 2010,This graphical language is suited for rapid application development. The MPC optimization problem is formulated in CVXGEN (http://cvxgen.com/) and the code generator provides a number of files written in plain C-code. The produced C-files are adapted slightly to make a library of this code. This is an easy way to call C-code from within the graphical programming language Lab-VIEW. The NI c Rio– 9024 Real-Time Controller is equipped with the Vx Works real-time operating system. CVXGEN code generator delivers very reliable code that can be implemented on a programmable automation controller system.

Reference [25] deals with a genetic algorithm implementation on a PLC and uses it for controller performance optimization. A Siemens TI505 PLC was programmed through TISOFT v. 4.3. The interface software, TISOFT in this case, allows the user to program loops (i.e., PID controllers), special function programs, along with some additional features as well as relay ladder logic GAs are a random search for an optimal solution; and, as such, their operation strongly relies on the availability of random numbers. Since random numbers are not available, a pseudo random number generator (PRNG) is implemented on the PLC in order to generate the necessary random actions. The objective of a PRNG is to generate a long sequence of numbers that are independent and uniformly distributed in [0, 1]. There are many different PRNGs among which the most used are linear congruently generators (LCGs). This is implemented on chemical plant having distillation column and reactors.

Motivation

As discussed in the above research papers and literature references, it is clear that not much work has been done for optimization on PLC platform. Also looking forward for practical aspects of small plants like solar, wind, electrical substations where harsh, humid, vibrations, temperatures are more prone to change, requires the rugged controller which can deal with advanced control algorithms.

So, there is a need of standalone optimization which is really efficient and effective. Almost all PLC deals with IEC 61131-3 standards for programming, but for advanced control applications it is not preferred because of complexity of code. So, an alternative approach for PLC programming along with future scope is discussed in this paper. The feasibility study checks that whether advanced control algorithms can be implemented on PLC. We will select the solvers for required algorithms in succeeding sections and will check the functionality of them.Scope of this project is to run the solver on the PLC and to check the functionality as well as feasibility of particular solver.

Why PLC for Advanced Optimization?

1)PLC is a dedicated system for controlling plant parameters so far they are used for conventional control like PID. APC algorithms are running on the PC platform; however PC has to do many tasks. In terms of ***Response speed*** PC based system is slow. There is a need of dedicated standalone system which can work for APC.

2) Being a ***Dedicated hardware*** if we can run those APC algorithms on PLC, less scanning time allows faster processing and more precise control.

3) PLC based system is more efficient because it provides ***Real time control*** to the system parameters.

4) The ***Communication latency*** between PLC and PC in conventional system can be removed by using PLC for advanced control.

5) Reliability, Robust structure, Resistance to Environment parameters, I/O handling capacity also enables PLC rather than a PC.

Introduction to Software & Programming

Today the PLCs are no longer limited to conventional control loops. Role of Programmable logic controllers has increased in modern industries to meet the performance requirements. To enhance the capabilities of higher end PLCs there are mostly two viewpoints considered. Comparative & Qualitative, in the first PLCs are compared with DCS or SCADA in terms of functionality and approach, whereas in second they are subjected to more advanced control algorithms in order to achieve better quality of control. The research described in this thesis is part of qualitative improvement of PLC.

IEC 61131-3 Standard for programming PLC is more complicated to write an embedded algorithm. Higher level language like C is having well developed libraries and techniques for multi-threaded and high reliability programming. C++ features can support better object oriented programming which generates even better results. Almost all the manufactures of PLC now supports a higher level language like C/C++ in to their PLC programming firmware to make it more align to embedded applications. Some of the popular manufactures are listed below.

- Rockwell Automation-Allen Bradley RS Logics 5000- C language support
- ABB AC-500 series PLC – C programming support
- Beck Hoff Automation-Twin cat 3-eXtended Automation (XA) Library-C/C++ language support
- Siemens C/C++ support
- GE Fanuc series 90-70/90-30 PLC – C programming tool kit
- B&R Automation PLCs- C support
- Mitsubishi PLC- C language support

In this thesis, ABB AC-500 series PLC, ETH-591 is considered for implementation purpose. The IDE used for programming in C is ABB PS501 Control Builder Plus [CBP] Version 2.3.0 which is a part of Automation Builder 1.0.1 Package. ABB Automation Builder is the integrated software suite for machine builders and system integrators wanting to automate their machines and systems in a productive way. Combining all of the tools required for configuring, programming, debugging and maintaining automation projects from a common intuitive interface, Automation Builder addresses the largest single cost element of most of today's industrial automation projects software.

The suite contains the programming and configuration tool for the ABB AC500 PLC, Control Builder Plus, with Drive Manager and drive application programming as integrated plug-in. additionally included are the Panel Builder for touch panel programming and the motion engineering tool, MINT Workbench. As a unique feature, ABB now includes the market leading offline robot programming and simulation tool, Robot Studio, providing customers with simple and fast interfacing of a PLC with a robot controller.

With the Automation Builder, various controller families can be combined in one automation project, and they can be configured and programmed from a single software suite. With a choice of various programming languages, the most suitable language for the particular application is available. At the same time, the USB-ROM distribution medium with a common installer ensures trouble-free and quick installation – on a customer-specific or project-specific basis.

The ABB Automation Builder contains the following components and functionalities for engineering AC500 PLCs, CP600 control panels, drives, motion control and robots:

- Installation manager
- Engineering platform
- Engineering of AC500, AC500-eCo and AC500-XC PLCs
- Configuration and programming of AC500-S Safety PLCs
- Human Machine Interface – CP600 control panels
- Configuration and programming of ACS drives
- Motion control – Micro Flex e150 servo drive
- Robot interface – IRC5 robot controller

Control Builder Plus is based on CoDeSys Automation Platform Technology by 3S-Smart software solutions.

For the implementation on PLC there are several approaches. For example MATLAB to PLC coder which generates structure text based on simple Simulink/MATLAB model, but it is not supported for complex functions like optimization. Also, MATLAB 2013 has features of C/C++ code generation but it also doesn't support conversion of higher ended algorithms. The same functionalities are available in SCILAB.

Using C-language based solvers for optimization is the most generic approach of implementation. It also checks the robustness and portability of any solver. We have considered some of the open source solvers as well as some commercial solvers but free for academic work.

Chapter-3

Screening of Solvers

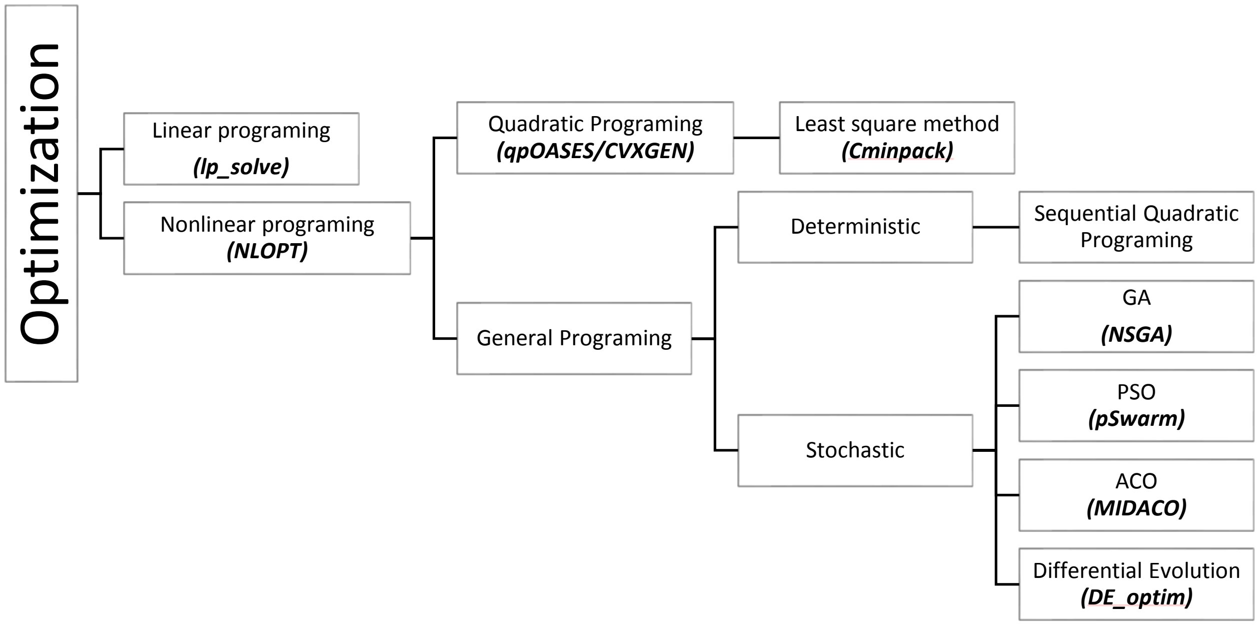


FIGURE 1 OPTIMIZATION HIERARCHY

General vs. Embedded solvers

• General solver (say, for QP)
  – Handles single problem instances with any dimensions, sparsely pattern
  – Typically optimized for large problems
  – Must deliver high accuracy
  – Variable execution time: stops when tolerance achieved

• Embedded solver
  – Solves many instances of the same problem family (dimensions, sparsely pattern) with different data
  – Solves small or smallish problems
  – Can deliver lower (application dependent) accuracy
  – Often must satisfy hard real-time deadline

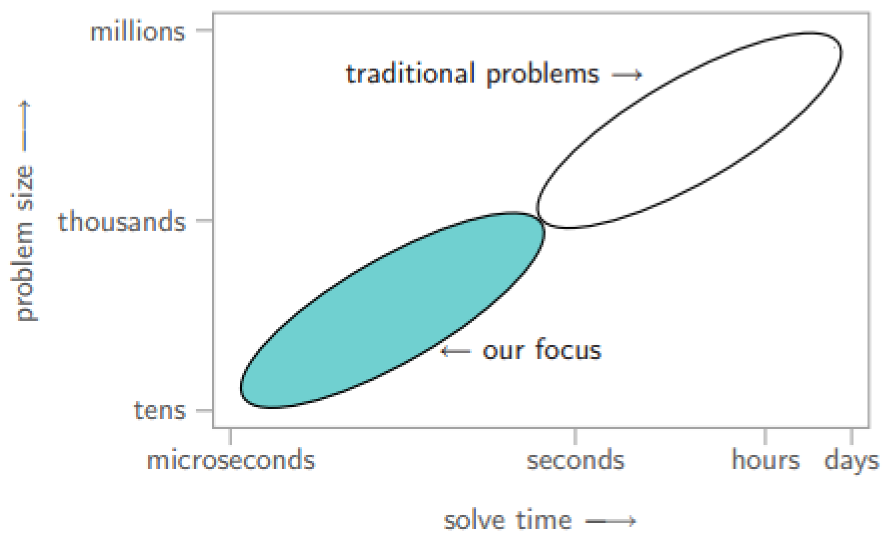
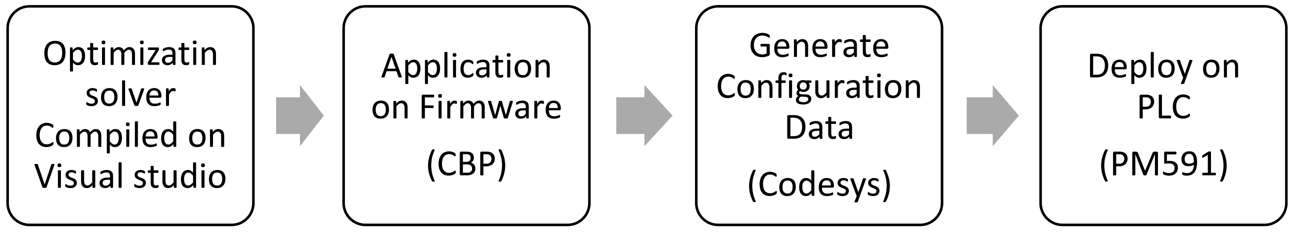


FIGURE 2 EMBEDDED OPTIMIZATION AT MILLISECOND/MICROSECOND TIME-SCALES

- Approach


Optimizatin solver Compiled on Visual studio
Application on Firmware (CBP)
Generate Configuration Data (Codesys)
Deploy on PLC (PM591)

The more robust and portable solvers are preferred while implementing on PLC. Some of the Optimization solvers are introduced bellow which gives idea about main features and type of problem that it solving, for more details please see the References.

Solvers Prescreening

Solvers	**Algoritham**	**Platform**
Cminpack	least squre method	C
MIDACO	Ant colony method	C/C++ Proprietary
Hildreth Solver	Hildrethalgoritham	C
qpOASES	Quadradic programing	C
CVXGEN	Online solver	C
MendelSolve	Genetic algorithm (GA) approach	Excel based solver
GNSO	several methods for global & nonsmooth problems (C/C++)	C/C++
Gagenes	genetic algorithm general solver	C
Knitro	nonlinear optimization	C/C++
CLP	linear programming	C++
Levmar	least square method	C++
NLOPT	Nonlinear optimization	C/C++
soplex	linear programming	C++
LIBQP	Convex Quadratic Programming	C
IBM ILOG CPLEX Optimizer	integer programing, linear programing,QP,convex,nonconvex	C Proprietary
GLPK	GNU Linear Programming Kit,MIP	C
SCIP	MIP(Solving Constraint Integer Programs)	C
AMPL	A Mathematical Programming Language	C
ALGLIB	all	C++Proprietary
IPOPT	Interior-Point Optimizer, for general large-scale nonlinear optimization	C
BONMIN	General MINLP Mixed Integer Nonlinear Programming	C++

splm	linear programming	C
LaGO	(Lagrangian Global Optimizer)global optimization of nonconvex mixed-integer nonlinear programs (MINLP)	C++
ROSE	Reformulation-Optimization Software Engine, software for performing symbolic reformulations to Mathematical Programs (MP)	proprietary(EXE)
MINOS	NLP	proprietary(EXE)
CONOPT	NLP	proprietary(EXE)
GUROBI	Linear and Mixed Integer Programming	Proprietary(EXE)
MOSEK	Linear and Mixed Integer Programming	Proprietary(EXE)
SDPA	Semidefinite programming (SDP)	-
CSDP	Semidefinite programming algorithm	C
PATH	Solving Mixed Complementarity Programming (MCP)	-
LGO	Lipschitz-continuous Global Optimizer	-
PSwarm	Hybrid derivative-free solver (particle swarm & direct search),	AMPL interface (C,Matlab)
MLOCPSOA	particle swarm paradigm, AMPL interface	(C source, binaries)
Black box opt	Black box functions and constraints, expensive evaluations, no derivatives needed	(C, GPL)
PGAPACK	Genetic algorithm	C, callable from Fortran
SPACE	global optimization through meta-modeling: fitting, cross -validation, prediction, minimization	C++
BARON	Branch-And-Reduce Optimization Navigator	Proprietary(EXE)

Solvers Selection Criteria

Selecting a solver for implementation depends on following criteria,

1) Solver size (must be less than 4 MB in the case of PM591)
2) Run time data space (must be less than)
3) There should not be any runtime argument in the main ( ) function.
4) All the errors on the PLC platform must be solvable
5) To check the algorithm feasibility there must be one solver from each category showed in the starting of this chapter

Depending on the above criteria we have selected 6 solvers which are described below. First the entire solver needs to be compiled on visual studio project for the consistency check and functionality check; then the entire package is transferred to control builder Plus and c code application is formed.

We need to remove all the portability codes in the given solver which are generally provided by the author to cover various environment. In order to implement on PLC the code must be very robust and most suitable for embedded control.

If solver size is more than 4 MB we need to do memory management to check the performance. All the 6 solvers are explained and the problem associated while implementing has been discussed.

It is also been taken care that selected solver has applications in different areas which can allow us to finalize the scope of application/case study. Visual studio 2010 compilation of the solver is explained as following.

qpOASES	Quadratic programming
Lp_solve	Linear Programming
PSwarm	Particle swarm intelligence & Pattern Search
MIDACO	Ant colony optimization
NSGA2	Genetic algorithm
Cminpack	Least square methods

qpOASES

qpOASES is an open-source c-implementation(Not Officially Release yet) of the recently proposed online active set strategy. It was inspired by important observations from the field of parametric quadratic programming and builds on the expectation that the optimal active set does not change much from one quadratic program to the next. It has several theoretical features that make it particularly suited for model predictive control applications. The software package qpOASES implements these ideas and also incorporates important modifications to make the algorithm numerically more robust. For more details refer [12].

$$\min_{x} \quad \frac{1}{2} x^T \, Hx + x^T \, g(w_0)$$
$$S.T. \quad lbA(w_0) \leq Ax \leq ubA(w_0)$$
$$lb(w_0) \leq x \leq ub(w_0)$$

Where, the Hessian matrix is symmetric and positive (semi-)definite and the gradient vector as well as the bound and constraint vectors depend affinely on the parameter w_0. It also can solve the Linear Programming by putting H matrix as zero.

Member [example] of the `QProblem` structure is created and using `QProblemCON( &example, intnV, intnC, Type of Hessian matrix)` problem dimensions are defined; problem initialized using `QProblem_init( &example,H,g,A,lb, ub,lbA,ubA,&nWSR,0)` function which returns SUCCESSFULL_RETURN or INIT_FAILED. On Successful return following functions can be used to solve the QP. Where, nV and nC represents No. of Variables and No. of Constrains respectively.

`QProblem_getPrimalSolution (&example,`X_{Opt}`);`

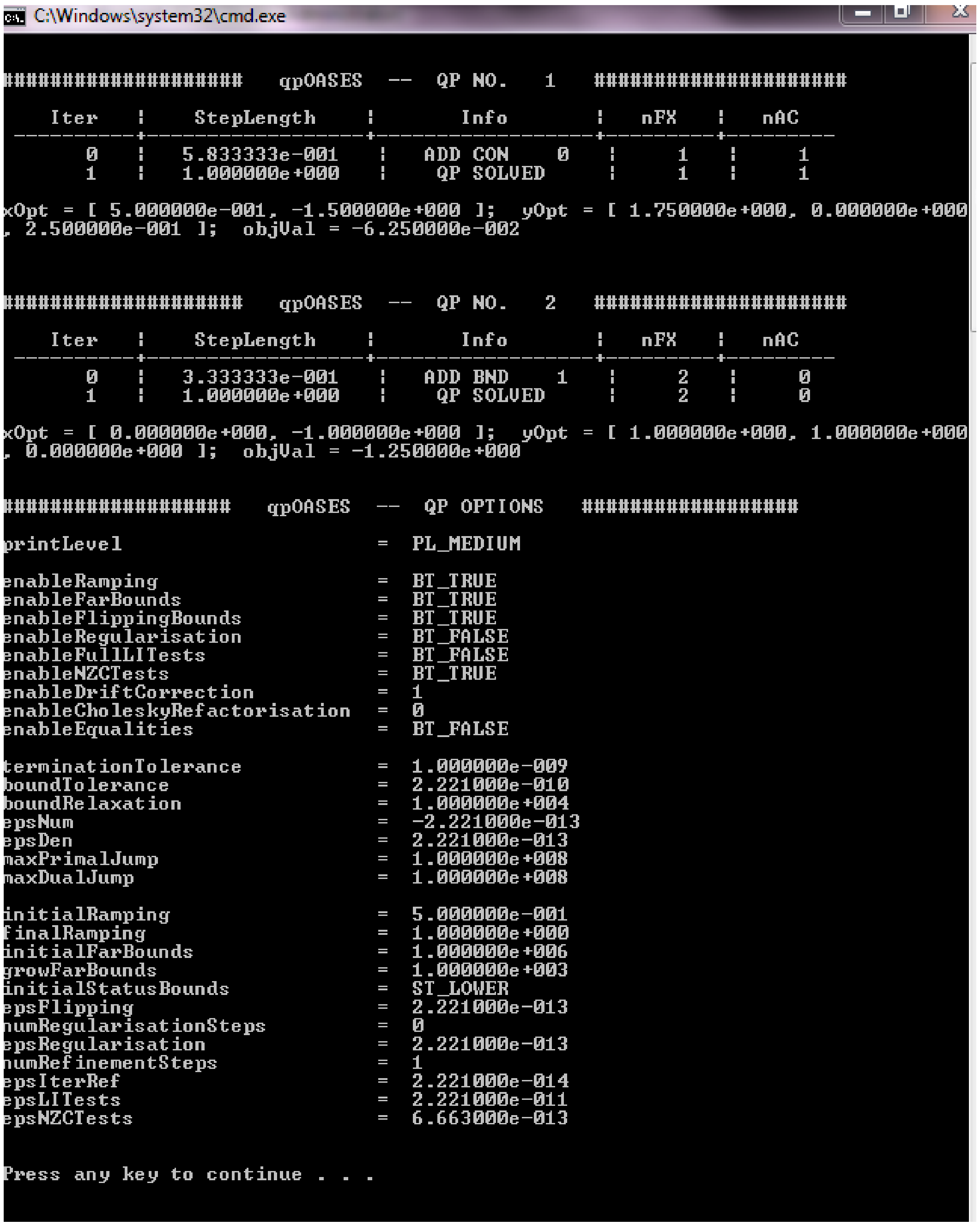
-that writes the optimal primal solution vector (dimension: nV) into the array X_{Opt}, which has to be allocated (and freed) by the user;

`QProblem_getDualSolution (&example,`Y_{Opt}`);`

-that writes the optimal dual solution vector2 (dimension: nV + nC) into the array Y_{Opt}, which has to be allocated (and freed) by the user;

`QProblem_getObjVal (&example);`

-that returns the optimal objective function value.



```
####################   qpOASES  --  QP NO.   1   ####################

    Iter   |    StepLength    |       Info       |   nFX    |   nAC
 ----------+------------------+------------------+----------+---------
       0   |   5.833333e-001  |   ADD CON    0   |     1    |     1
       1   |   1.000000e+000  |    QP SOLVED     |     1    |     1

xOpt  =  [ 5.000000e-001, -1.500000e+000 ];  yOpt  =  [ 1.750000e+000, 0.000000e+000
, 2.500000e-001 ];  objVal = -6.250000e-002


####################   qpOASES  --  QP NO.   2   ####################

    Iter   |    StepLength    |       Info       |   nFX    |   nAC
 ----------+------------------+------------------+----------+---------
       0   |   3.333333e-001  |   ADD BND    1   |     2    |     0
       1   |   1.000000e+000  |    QP SOLVED     |     2    |     0

xOpt  =  [ 0.000000e+000, -1.000000e+000 ];  yOpt  =  [ 1.000000e+000, 1.000000e+000
, 0.000000e+000 ];  objVal = -1.250000e+000


###################   qpOASES  --  QP OPTIONS   ##################

printLevel                     =  PL_MEDIUM

enableRamping                  =  BT_TRUE
enableFarBounds                =  BT_TRUE
enableFlippingBounds           =  BT_TRUE
enableRegularisation           =  BT_FALSE
enableFullLITests              =  BT_FALSE
enableNZCTests                 =  BT_TRUE
enableDriftCorrection          =  1
enableCholeskyRefactorisation  =  0
enableEqualities               =  BT_FALSE

terminationTolerance           =  1.000000e-009
boundTolerance                 =  2.221000e-010
boundRelaxation                =  1.000000e+004
epsNum                         =  -2.221000e-013
epsDen                         =  2.221000e-013
maxPrimalJump                  =  1.000000e+008
maxDualJump                    =  1.000000e+008

initialRamping                 =  5.000000e-001
finalRamping                   =  1.000000e+000
initialFarBounds               =  1.000000e+006
growFarBounds                  =  1.000000e+003
initialStatusBounds            =  ST_LOWER
epsFlipping                    =  2.221000e-013
numRegularisationSteps         =  0
epsRegularisation              =  2.221000e-013
numRefinementSteps             =  1
epsIterRef                     =  2.221000e-014
epsLITests                     =  2.221000e-011
epsNZCTests                    =  6.663000e-013


Press any key to continue . . .
```

FIGURE 3 QPOASES ON VISUAL STUDIO

qpOASES is the highly efficient solver for quadratic programming and specifically Model Predictive Control Applications. Predictive Engine Control, Emission control of integral gas engines, MPC of a Diesel engine test bench at University of Linz, Austria, on dSPACE hardware at sampling times of 50 millisecondsApplications to Predictive Engine Control, etc.

Lp_solve

LP_solve is the open source C-based linear programming solver available under GNU Lesser General Public License. Compared to Gurobi, CPLEX, SCIP, it is not the best and fastest solver for the linear programming but as a part of research it is considered because of its portability and open source code. LP_solve can also solve MILP problems. Some of the functions of this toolkit which describe the overall function of the LP_solve are described below.

`lp1=make_lp(nV,nC);`
-creates new problem which has nV and nC; nV=No. of variables, nC No. of Constrains.

`set_maxim(lp1); or set_minim(lp1);`
-set maxim for Maximization problem and set minim for Minimization problem

`str_add_constraint(lp1," ",Constrain type, RHS);`
-will add the constraints in specified LP, and supports GE (Greater then), LE(Less then), EQ (Equality constrains) constrains types followed by right-hand side values.

`str_set_obj_fn (lp1, "");`
-will set the objective function coefficients for the specified LP

`Solve (lp1)`
-Returns the value zero if the algorithm has found an optimal value.

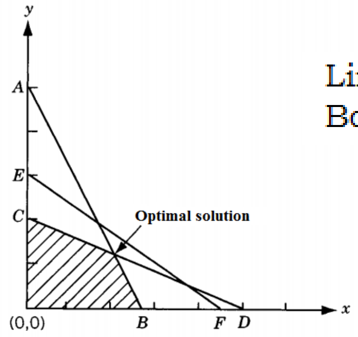
`set_mat (lp1, Row, Column, 0.5);`
-Single element of the matrix can be changed by this command crated by intersection of specified Row and Column.

Example:

- A manufacturing firm produces two machine parts using lathes, milling machine and grinding machines. The different machining time required for each part, the machining times available on different machines, and the profit on each machine part are given in the following table. Determine the No of parts I and II manufactured per week to *maximize the profit*.

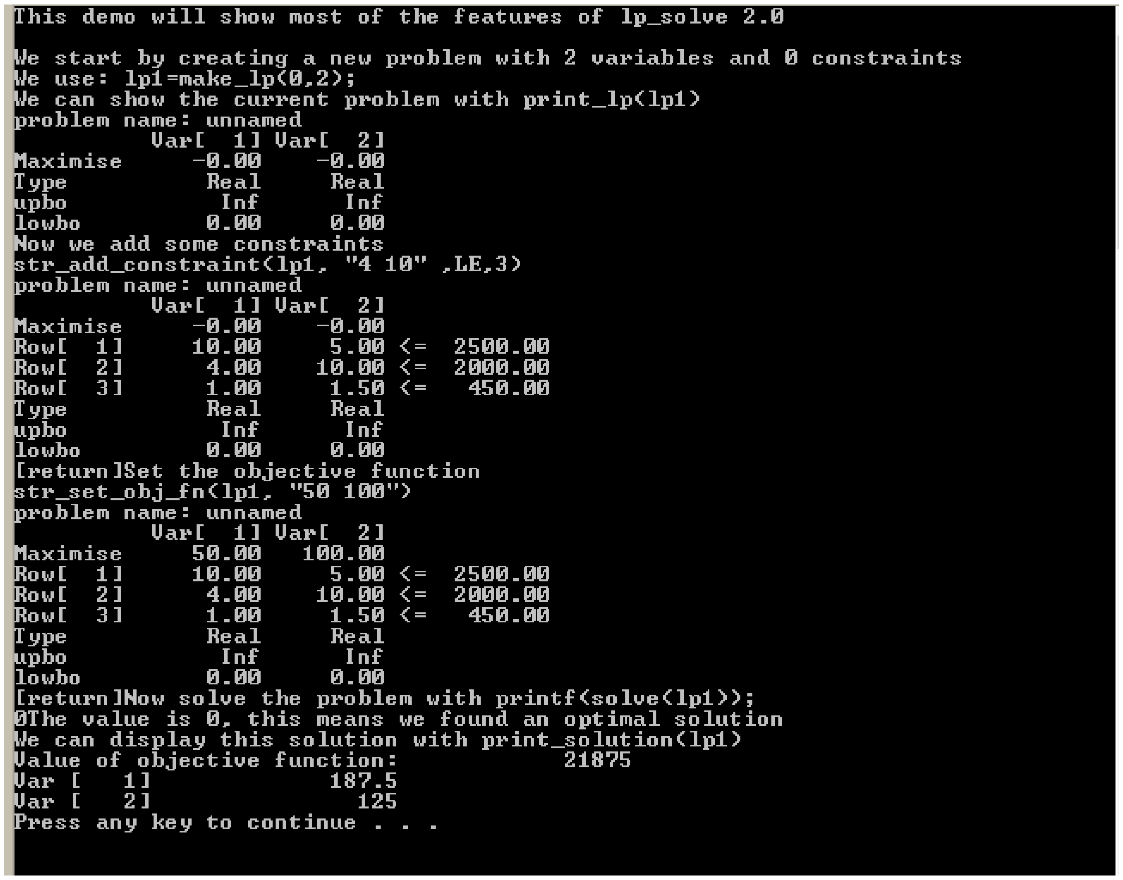
Type of machine	Machine part1(machining time in min)	Machine part2(machining time in min)	Max time available per week(min)
Lathe	10	5	2500
Milling machine	4	10	2000
Grinding machine	1	1.5	450
Profit_per_Unit	$50	$100	

Objective: Maximize (x, y) = 50x+ 100y, Subject to following constraints, 10x + 5y <= 2500, 4x + 10y <= 2000, x + 1.5y <=450, x>=0, y>=0



Linear Programming Sollution at the Boundry of the constraints Space

(187.5, 125)


```
This demo will show most of the features of lp_solve 2.0

We start by creating a new problem with 2 variables and 0 constraints
We use: lp1=make_lp(0,2);
We can show the current problem with print_lp(lp1)
problem name: unnamed
          Var[  1] Var[  2]
Maximise     -0.00    -0.00
Type          Real     Real
upbo           Inf      Inf
lowbo         0.00     0.00
Now we add some constraints
str_add_constraint(lp1, "4 10" ,LE,3)
problem name: unnamed
          Var[  1] Var[  2]
Maximise     -0.00    -0.00
Row[  1]     10.00     5.00 <=  2500.00
Row[  2]      4.00    10.00 <=  2000.00
Row[  3]      1.00     1.50 <=   450.00
Type          Real     Real
upbo           Inf      Inf
lowbo         0.00     0.00
[return]Set the objective function
str_set_obj_fn(lp1, "50 100")
problem name: unnamed
          Var[  1] Var[  2]
Maximise     50.00   100.00
Row[  1]     10.00     5.00 <=  2500.00
Row[  2]      4.00    10.00 <=  2000.00
Row[  3]      1.00     1.50 <=   450.00
Type          Real     Real
upbo           Inf      Inf
lowbo         0.00     0.00
[return]Now solve the problem with printf(solve(lp1));
0The value is 0, this means we found an optimal solution
We can display this solution with print_solution(lp1)
Value of objective function:            21875
Var [    1]               187.5
Var [    2]                 125
Press any key to continue . . .
```

FIGURE 4 LPSOLVE ON VISUAL STUDIO

Particle swarm optimization (**PSO**) is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. PSO optimizes a problem by having a population of candidate solutions, here dubbed particles, and moving these particles around in the search-space according to simple mathematical formula over the particle's position and velocity. Each particle's movement is influenced by its local best known position but, is also guided toward the best known positions in the search-space, which are updated as better positions are found by other particles. This is expected to move the swarm toward the best solutions.

PSwarm is the Hybrid Solver for Linearly Constrained Global Derivative-Free Optimization based on Particle swarm pattern search algorithm available on GNU Lesser General Public License. It uses following functions.it solves problems defined by general linear constraints (without using the derivatives of the objective function, which might be non-smooth and/or noisy). We treat only the case of inequality constraints. (Equality constraints can be converted into inequalities, although the process is known to introduce degeneracy.) It basically solves these types of problems. For more details refer [30].

$$\min_{Z \in \Omega} f(z)$$
$$S.T. \quad Az \leq b$$
$$\Omega = \{z \in R^n : \ l \leq z \leq u\}$$

`set_problem_dimension (&n, &lincons);`

-It initialize the problem with n variables and lincons no of linear constrains; no. of objective function coefficient is equal to lincons.

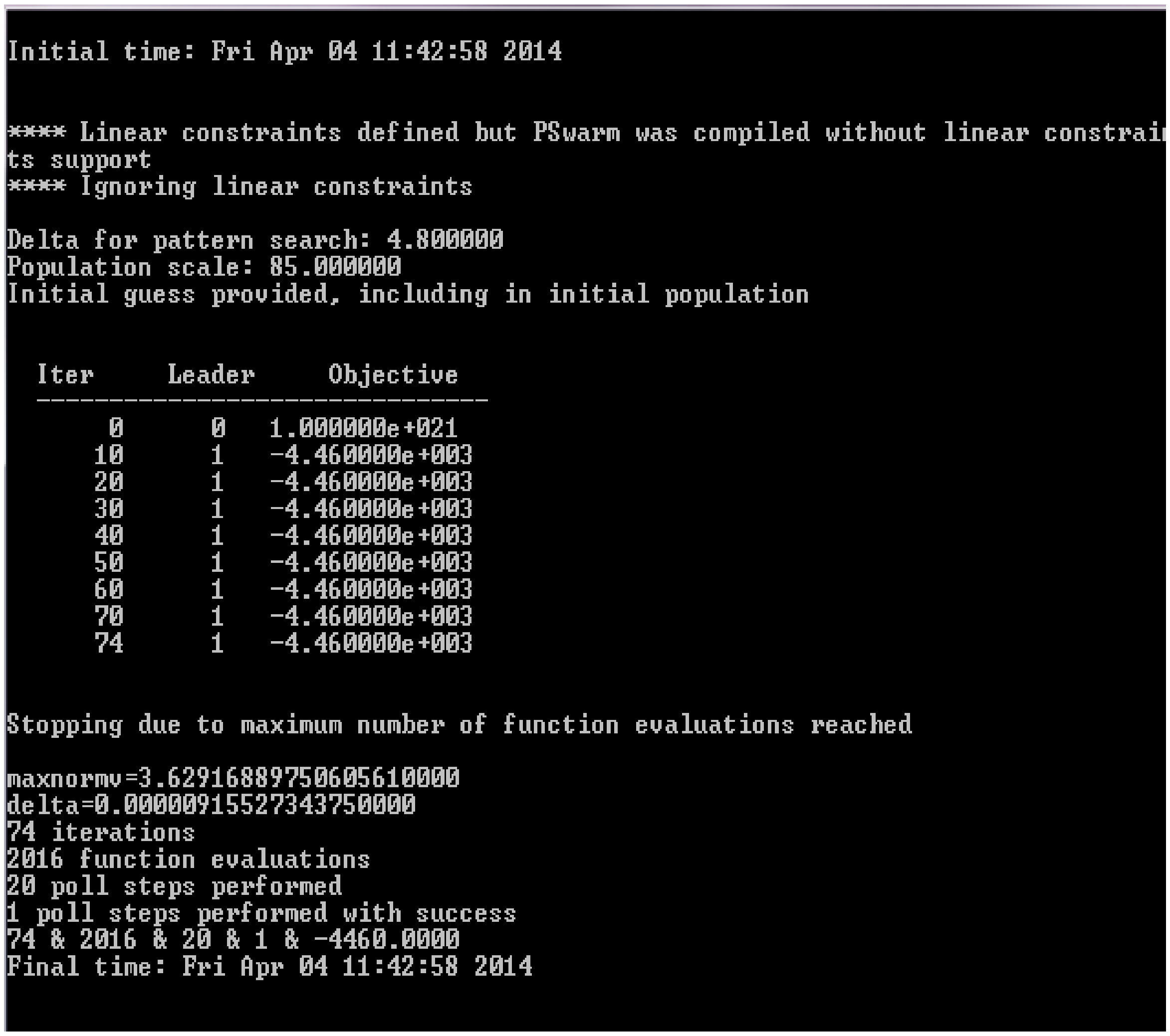
`set_problem(X₀, lb, ub, A, b);`

-X_0 is the initial guess provided by the user, lb is lower bound array and ub is upper bound array, A, b are constraints specific matrices.

`user_init ();`

-This functions allows to set the objective function co-efficient by default.

`PSwarm (n, &objfun, lb, ub, lincons, A, b, &sol, f, X0);`-It starts the random number generator andwhich seeds the random solutions with time; the algorithm will stop when maximum iterations or maximum objective function evolutions will reach.



```
Initial time: Fri Apr 04 11:42:58 2014


**** Linear constraints defined but PSwarm was compiled without linear constrai
ts support
**** Ignoring linear constraints

Delta for pattern search: 4.800000
Population scale: 85.000000
Initial guess provided, including in initial population


  Iter     Leader     Objective
  -------------------------------
      0      0   1.000000e+021
     10      1   -4.460000e+003
     20      1   -4.460000e+003
     30      1   -4.460000e+003
     40      1   -4.460000e+003
     50      1   -4.460000e+003
     60      1   -4.460000e+003
     70      1   -4.460000e+003
     74      1   -4.460000e+003


Stopping due to maximum number of function evaluations reached

maxnormv=3.62916889750605610000
delta=0.00000915527343750000
74 iterations
2016 function evaluations
20 poll steps performed
1 poll steps performed with success
74 & 2016 & 20 & 1 & -4460.0000
Final time: Fri Apr 04 11:42:58 2014
```

FIGURE 5 PSWARM ON VISUAL STUDIO

As shown in the above screenshot, the given solver solves the problem in order of milliseconds and minimizes the objective function value from pattern search and swarm intelligence method.

PSwarm has been used in various areas like adaptive control, power system optimization problems**,**Reactive Power and Voltage ControlCapacitor and FACTS Placement**,**Controller Tuning**,**Short-Term Load Forecasting (STLF) Generator Contributions to Transmission System**.**

In the natural world, ants (initially) wander randomly, and upon finding food return to their colony while laying down pheromone trails. If other ants find such a path, they are likely not to keep travelling at random, but to instead follow the trail, returning and reinforcing it if they eventually find food.

Over time, however, the pheromone trail starts to evaporate, thus reducing its attractive strength. The more time it takes for an ant to travel down the path and back again, the more time the pheromones have to evaporate. A short path, by comparison, gets marched over more frequently, and thus the pheromone density becomes higher on shorter paths than longer ones. Pheromone evaporation also has the advantage of avoiding the convergence to a locally optimal solution.

If there were no evaporation at all, the paths chosen by the first ants would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained. Thus, when one ant finds a good (i.e., short) path from the colony to a food source, other ants are more likely to follow that path, and positive feedback eventually leads to all the ants' following a single path. The idea of the ant colony algorithm is to mimic this behavior with "simulated ants" walking around the graph representing the problem to solve.

MIDACO is commercialgeneral-purpose software for solving mathematical optimization problems. The software implements an **extended ant colony optimization** algorithm, which is a heuristic method that stochastically approximates a solution to the mathematical problem. MIDACO can be applied on purely continuous (NLP), purely combinatorial (IP) or mixed integer nonlinear programming (MINLP) optimization problems. Problems may be restricted to nonlinear equality and/or inequality constraints. The objective and constraint functions are treated as black box by MIDACO.

The MIDACO algorithm is based on an evolutionary Metaheuristic called Ant Colony Optimization (ACO), which was extended to the mixed integer search domain in [34]. The ACO algorithm inMIDACO is based on a so called multi-kernel Gaussian probability density functions (PDF), whichgenerate samples of iterates (called Ants). It solves problems type of following.

$$\boldsymbol{Minimize\ F(x,y)\ (x \in R^{n_{con}}, y \in Z^{n_{int}}, n_{con}, n_{int} \in N)}$$

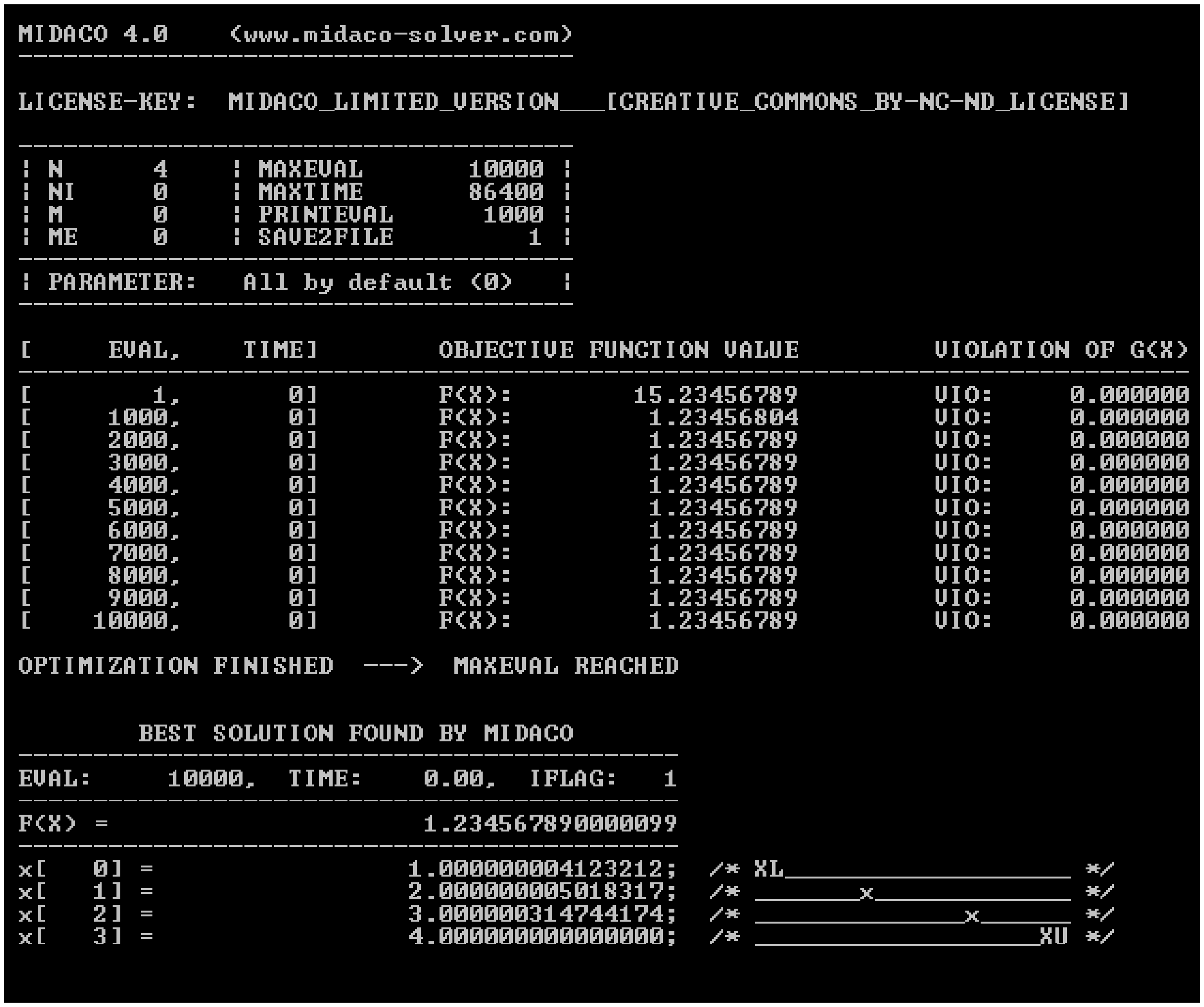
$$\begin{aligned} \textit{subject to}: \quad & g_i(x,y) = 0, & i = 1, \ldots m_e \in N \\ & g_i(x,y) \geq 0, & i = m_e + 1, \ldots, m \in N \\ & x_i \leq x \leq x_u, \quad x_i, x_u \in R^{n_{con}}, & y_i \leq y \leq y_u, \quad y_i, y_u \in N^{n_{int}} \end{aligned}$$

MIDACO as a solver requires some of the parameters to be specified in order to solve the problem. These parameters are given with details in reference [33]. Some of the functions of this solver are specified below.

```
MIDACO (P, N, NI, M, ME, X, F, G, XL, XU, IFLAG, ISTOP, PARAM, RW,
LRW, IW, LIW, KEY)

Where,
P: Parallelization factor
N: Number of optimization variables in total (continuous and integer
ones)
NI: Number of integer optimization variables. 'NI' <= 'N'
M:  Number of constraints in total (equality and inequality ones)
ME: Number of equality constraints. 'ME' <= 'M'.
X (P*N): Array containing theiterate 'X'.
XL (N): Array containing the lower bounds for the iterates 'X'
XU (N): Array containing the upper bounds for the iterates 'X'
IFLAG: Information flag used by MIDACO to indicate whether the
operation is successfully completed or with errors & warnings.
ISTOP:   Communication flag to stop MIDACO
PARAM: Parameter specifications to adjust the Accuracy, Random
seeds, etc.
RW (LRW):  Real work array (Type: double precision) of length 'LRW'
LRW: Length of 'RW'. 'LRW' must be greater or equal to 200*N+2*M+1000
IW (LIW): Integer work array (Type: long integer) of length 'LIW'
LIW: Length of 'IW'. 'LIW' must be greater or equal to 2*N+P+1000
KEY:  License-Key for MIDACO.
```

MIDACO does provide four different stopping criteria: MAXTIME, MAXEVAL, AUTOSTOP and FSTOP. MAXTIME specifies a maximum CPU time budget (e.g. 60 Seconds) during which MIDACO is allowed to perform its search process. Analogue to MAXTIME, the MAXEVAL criteria specifies a maximum number of function evaluations (e.g. 1000). If AUTOSTOP is active, MIDACO will stop automatically by itself. FSTOP defines a specific value for the objective function F(X) to be reached.



```
MIDACO 4.0    (www.midaco-solver.com)
--------------------------------------

LICENSE-KEY:  MIDACO_LIMITED_VERSION___[CREATIVE_COMMONS_BY-NC-ND_LICENSE]

--------------------------------------
! N      4    ! MAXEVAL       10000 !
! NI     0    ! MAXTIME       86400 !
! M      0    ! PRINTEVAL      1000 !
! ME     0    ! SAVE2FILE         1 !
--------------------------------------
! PARAMETER:   All by default (0)   !
--------------------------------------

[     EVAL,    TIME]       OBJECTIVE FUNCTION VALUE         VIOLATION OF G(X)
------------------------------------------------------------------------------
[        1,       0]       F(X):       15.23456789          VIO:     0.000000
[     1000,       0]       F(X):        1.23456804          VIO:     0.000000
[     2000,       0]       F(X):        1.23456789          VIO:     0.000000
[     3000,       0]       F(X):        1.23456789          VIO:     0.000000
[     4000,       0]       F(X):        1.23456789          VIO:     0.000000
[     5000,       0]       F(X):        1.23456789          VIO:     0.000000
[     6000,       0]       F(X):        1.23456789          VIO:     0.000000
[     7000,       0]       F(X):        1.23456789          VIO:     0.000000
[     8000,       0]       F(X):        1.23456789          VIO:     0.000000
[     9000,       0]       F(X):        1.23456789          VIO:     0.000000
[    10000,       0]       F(X):        1.23456789          VIO:     0.000000

OPTIMIZATION FINISHED  --->  MAXEVAL REACHED


        BEST SOLUTION FOUND BY MIDACO
---------------------------------------------
EVAL:     10000,  TIME:    0.00,  IFLAG:   1
---------------------------------------------
F(X) =                       1.234567890000099
---------------------------------------------
x[   0] =                  1.000000004123212;  /* XL_____________________ */
x[   1] =                  2.000000005018317;  /* ________x______________ */
x[   2] =                  3.000000314744174;  /* ________________x______ */
x[   3] =                  4.000000000000000;  /* _____________________XU */
```

FIGURE 6 MIDACO SOLVED ON VISUAL STUDIO

MIDACO solves some of the benchmarking problems in different domains Satellite Constellation, Tennessee Eastman Process (TEP), Cooperative Wireless Networks, Travelling sales man problem, Optimal Camera Placement, Space Launch Vehicle (Boeing Delta III) and Structural Optimization of Submarine Hulls. For more details on Applications refer [33].

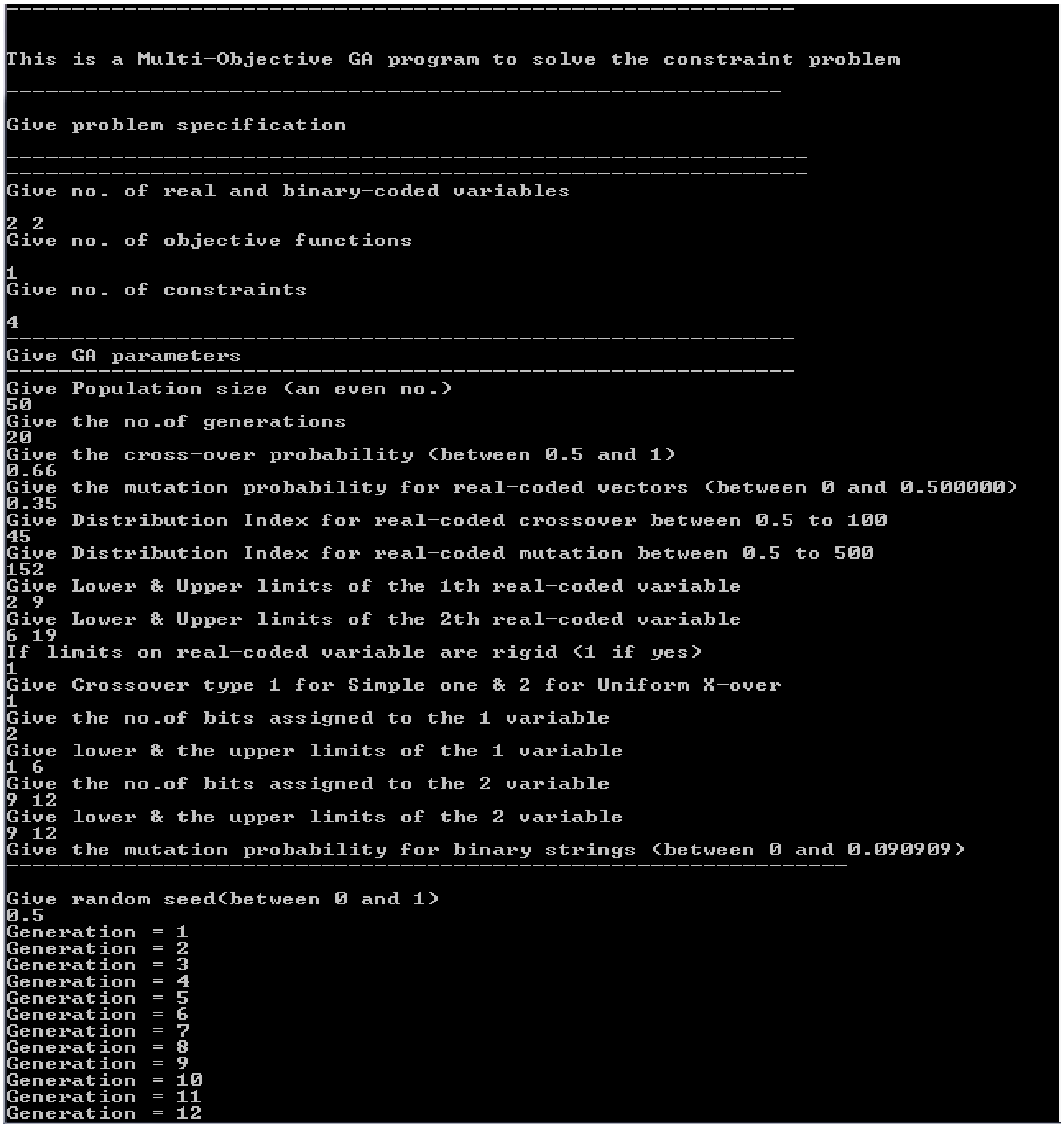
In a genetic algorithm, a population of candidate solutions (called individuals, creatures, or phenotypes) to an optimization problem is evolved toward better solutions. Each candidate solution has a set of properties (its chromosomes or genotype) which can be mutated and altered.

The evolution usually starts from a population of randomly generated individuals, and is an iterative process, with the population in each iteration called a generation. In each generation, the fitness of every individual in the population is evaluated; the fitness is usually the value of the objective function in the optimization problem being solved. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new generation. The new generation of candidate solutions is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population.

NSGA2 is the Multi-objective genetic algorithm to solve the constraint problem solver available under open source software solutions.Program asks for GA parameters like Population Size,Chromosome Length,No. of generations,No. of Functions, No. of Constraints, No. of binary-coded variables,No. of real-coded variables,Cross-over Probability,Mutation Probability etc.Random Number Generator generates random numbers which are used to initialized population in the constrained domain which can be treated as first generation.

new_pop_ptr and old_pop_ptr stores pointer reference to new and old population generations respectively. mate_pop_ptr is the pointer which stores some of best solutions for succeeding generation depending on selection criteria from selection algorithm to get the different individuals to be selected.

After the mutation and crossover routine gets formulated, the evaluation of the generation will be done by `void func(population *pop_ptr)`function. This functioncontains in file func-con.h which latch the fitness function from the main subroutine and constraints violations conditions. This process runs in loop until the maximum evaluation occurs.



```
------------------------------------------------------------

This is a Multi-Objective GA program to solve the constraint problem
-----------------------------------------------------------

Give problem specification

-------------------------------------------------------------
-------------------------------------------------------------
Give no. of real and binary-coded variables

2 2
Give no. of objective functions

1
Give no. of constraints

4
------------------------------------------------------------
Give GA parameters
------------------------------------------------------------
Give Population size (an even no.)
50
Give the no.of generations
20
Give the cross-over probability (between 0.5 and 1)
0.66
Give the mutation probability for real-coded vectors (between 0 and 0.500000)
0.35
Give Distribution Index for real-coded crossover between 0.5 to 100
45
Give Distribution Index for real-coded mutation between 0.5 to 500
152
Give Lower & Upper limits of the 1th real-coded variable
2 9
Give Lower & Upper limits of the 2th real-coded variable
6 19
If limits on real-coded variable are rigid (1 if yes)
1
Give Crossover type 1 for Simple one & 2 for Uniform X-over
1
Give the no.of bits assigned to the 1 variable
2
Give lower & the upper limits of the 1 variable
1 6
Give the no.of bits assigned to the 2 variable
9 12
Give lower & the upper limits of the 2 variable
9 12
Give the mutation probability for binary strings (between 0 and 0.090909)
----------------------------------------------------------------

Give random seed(between 0 and 1)
0.5
Generation = 1
Generation = 2
Generation = 3
Generation = 4
Generation = 5
Generation = 6
Generation = 7
Generation = 8
Generation = 9
Generation = 10
Generation = 11
Generation = 12
```

FIGURE 7 NSGA2 ON VISUAL STUDIO

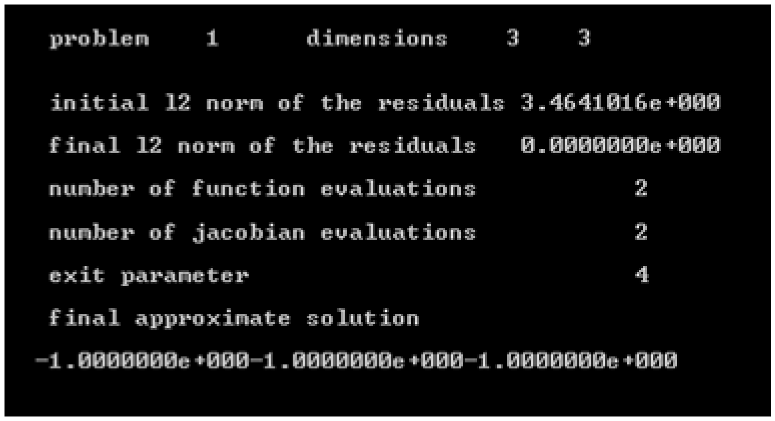
The size of this code is 6 MB whereas the limit for PLC is 4 MB, so for this algorithm we need to change the solver. There are other genetic algorithm solvers which requires less size.

C version of the minpack minimization package. It has been derived from the FORTRAN code using f2c and some limited manual editing. The method of least squares is a standard approach to the approximate solution of over determined systems, i.e., sets of equations in which there are more equations than unknowns. "Least squares" means that the overall solution minimizes the sum of the squares of the errors made in the results of every single equation.

Minpack includes software for solving nonlinear equations and nonlinear least squares problems. Five algorithmic paths each include a core subroutine and an easy-to-use driver. The algorithms proceed either from an analytic specification of the Jacobian matrix or directly from the problem functions. The paths include facilities for systems of equations with a banded Jacobian matrix, for least squares problems with a large amount of data, and for checking the consistency of the Jacobian matrix with the functions. One of the functions which have been used for implementation is described below.

LMDIF Function:

The purpose of LMDIF is to minimize the sum of the squares of M nonlinear functions in N variables by a modification of the Levenberg-Marquardt algorithm. The user must provide a subroutine which calculates the functions. The Jacobean is then calculated by a forward-difference approximation.



```
  problem    1      dimensions    3    3

  initial l2 norm of the residuals 3.4641016e+000
  final l2 norm of the residuals   0.0000000e+000
  number of function evaluations           2
  number of jacobian evaluations           2
  exit parameter                           4
  final approximate solution
 -1.0000000e+000-1.0000000e+000-1.0000000e+000
```

FIGURE 8 LEAST SQUARE MINMIZATION OF FUNCTION

For more details refer [35]

Summary

This summary includes challenges while implementation and issues which are resolved during this project so far.

1) A crucial part of the preparation process of qpOASESis the cross-build/link stage which brings out some issues. Relocation in ".data" segment is not supported, and the (amount of) initialized static/global variables cause problems at the linking stage. Specifically, the linker could not handle the huge "return_Value_List" array in "MessageHandling.c", and also "TRANS, NOTRANS" initializations in "Matrices.c". This problem is also found in Pswarm.

2) Runt time arguments which are required to be supplied needs to be removed and replaced with arrays.

3)"Size of the data is too high" error represents that solver requires more memory then PLC has. In this case we need to use another PLC which can have more memory for run time data as well as program memory or we need to go for memory management strategies.

4) Printf and Scanfwhich are used in c code requires to be removed or replaced with control builder plus I/O interfacing.

5) To view the run time data allocation size the command used in control builder plus is –fstake –usage which generates the excel compatibility file and the data allocated to each variable can be obtained.

6) We need to change the Calloc, Realloc functions which represents dynamic allocation because some of this functions are not supported.

7) Struct Redefinition errors are resolved for lp_solve by removing some of the included header.

Chapter-4

Functional Testing of Selected Solver

The Solver which is used to compile on PLC is Cminpack. It uses lmdif function to check the 3 variable with 3 equations. M is a positive integer input variable set to the number of functions. N is a positive integer input variable set to the number of variables which must not exceed M. For the following libraries are assumed to be declared.

```
void POU(POUtyp* inst)
{
int m, n, info, lwa, iwa[3], one=1;
lwa = 1000;
doubletol, fnorm, x[3], fvec[3], wa[1000];

  m = 3;
  n = 3;

/* the following starting values provide a rough fit. */

x[0] = 1.e0;
x[1] = 1.e0;
x[2] = 1.e0;

x[0] = 0.5;
x[1] = 0.5;
x[2] = 0.5;


/* set tol to the square root of the machine precision. unless high precision
solutions are required, this is the recommended  setting. */

tol = sqrt(dpmpar_(&one));

  lmdif1_ (&fcn1, &m, &n, x, fvec, &tol, &info, iwa, wa, &lwa);

fnorm = enorm_(&m, fvec);

inst->Y1 = x[0];
inst->Y2 = x[1];
inst->Y3 = x[2];

  }

void fcn1(constint *m, constint *n, constdouble *x, double *fvec, int *iflag)
{
```

```
/* function fcn for lmdif1 example */

       double c12, c13, c23, a1, a2, a3;

       c12 = 0.3265;
       c13 = 0.16;
       c23 = 0.3265;

       a1 = x[0];
       a2 = x[1];
       a3 = x[2];

       fvec[0] = 4*a1*pow((1-a1), 2) + 4*a2*pow((1-a2), 2) * pow((1 - 2*a1*c12), 3) +
4*a3*pow((1-a3), 2) * pow((1 - 2*pow((pow((a1*c13),2) + pow((a2*c23),2)),0.5)),3);

       if (abs(fvec[0]) > 1e-6)
             fvec[0] = 1/fvec[0];
       else
             fvec[0] = 1e10;

       fvec[1] = 0;
       fvec[2] = 0;
}
```

FVEC is an output array of length M which contains the functions evaluated at the output X. The function specified above can be run on PLC and PC both. We cannot compare the time directly as PC and PLC has different Processors which run on different clocks. The detailed comparison of PC vs. PLC can be considered as a part of future scope.

In computing, FLOPS (for Floating-point Operations per Second) is a measure of computer performance, useful in fields of scientific calculations that make heavy use of floating-point calculations. For such cases it is a more accurate measure than the generic instructions per second.

For now we have time required to solve the problem in PC (when PC is loaded with other programs) is 121 milliseconds. Whereas on PLC it is less than(68 msec) this value but the data packaging and linking stage modifies a code, so the actual comparison in terms of FLOPS is considered as part of future investigation.

But it is concluded that Optimization can be done on PLC, Our future investigation will deals with implementation of other solvers as well as comparison of computational parameters.

From this experiment the nature of Deterministic and non- Deterministic computation can be observed.

Potential Applications

PLC based optimization has scope of implementation in different areas; some of them are described as an Optimization Problem.

- HVAC systems
- Transformer Optimal thermal regulation
- Intelligent Building control
- Smart Irrigation Farm control
- Irrigation Network control
- Wind/Hydro turbine Control Compressor control + Anti surge control
- Waste Water treatment
- Polymerization Reactor (Extruder)
- Furnace Control Systems (Temperature Control, Pressure Control)
- Solar heating & Power generation using PV
- Power quality Management for Renewable energy plants
- CNC/Lathe machines
- Chemical Mixer
- Diesel engine Emission control
- Humidifier & Dehumidifier
- Hydraulic positioning or force control on the wet and dry ends of paper machines
- Robotics as a part of Motion Control application
- Speed control of DC/AC motors
- Control of planar machine
- Automatic frequency control of Induction heating
- Pharmaceutical Batch Mixing processes
- Machine Control (Pump Control Systems, Valve Control Systems)
- Cement plant (Grinding mill)
- Power Plant Emission control (Power plant ESP - Electro Static Precipitator)
- Boiler & Burner optimal Sequence control.
- Generator Excitation control
- Automatic Welding machine or Paint shop of Automobile manufacturing system
- Loading Arms of Crude Oil Ships for optimum positioning
- AGV-Automatically guided vehicle for optimal path trajectory

HVAC Systems

HVAC= Heating ventilating Air conditioning or High volume Air-conditioning

Rooms with Common air Header

Control
- Temperature
- Concentration- CO_2
- Humidity

Objective:

- To distribute the Load on Compressors
- To control the Atmosphere of Building depending on occupancy

→Current Strategies: PID on PLC
MPC on OPC-server

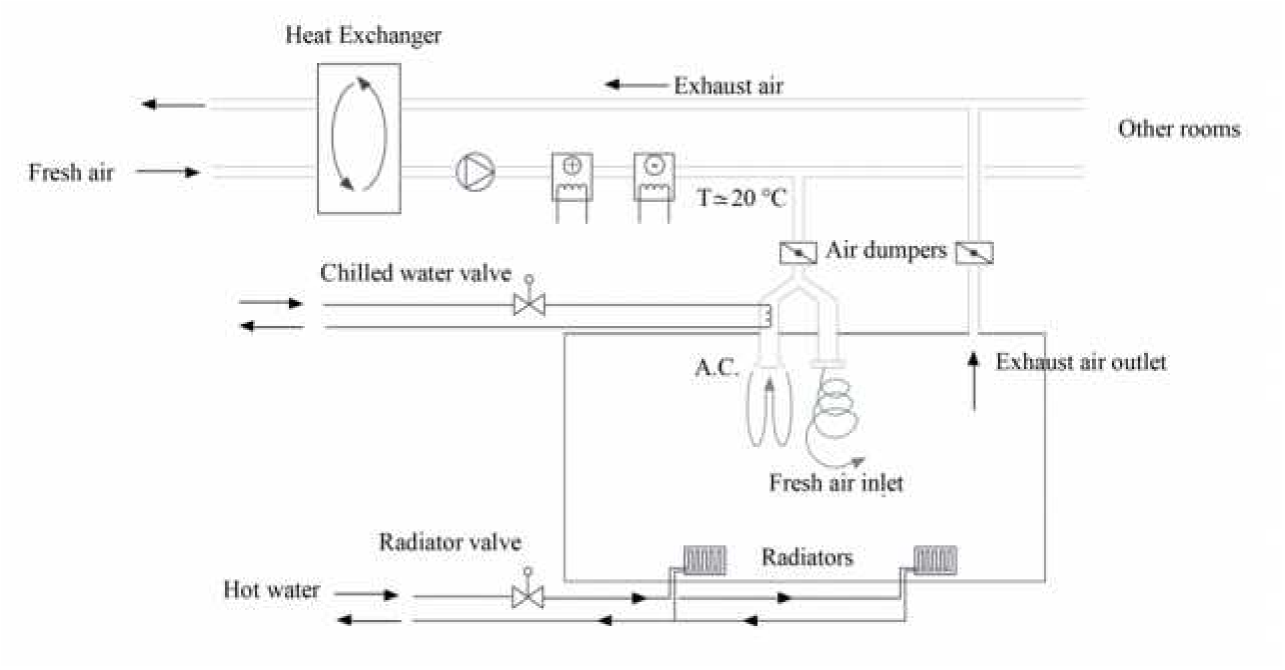


FIGURE 91 TOPOLOGY OF HVAC SYSTEMS

The PLCs can be used to decide the optimized set points of Temperature, Humidity and Ventilation level along with real time weather measurements. We can mathematically model the parameters which indicate the comforts along with energy requirements in terms of heating, Air-conditioning, ventilation etc. and apply optimization algorithms to minimize the cost function. Ultimately, we can try to achieve an efficient HVAC system without compromising our comforts. There are parameters which defines comforts. This strategy can be applicable to Data center cooling as well as Intelligent Building Control.

Chapter-5

Conclusion, Remarks & Future Scope

During this research the challenging part of putting highly computational algorithms on PLC is limitations on memory and processing power. As higher end PLCs of almost all the vendors having enough memory. So, it enables to compute optimization problems.

In this report the generic approach for implementing c code algorithm on PLC is discussed along with its application areas. We also discussed why PLC for advanced control & optimization. Almost for all optimization algorithms we can dedicate one solver and that can be implemented on PLC to enhance the capabilities, some test problems can be solved and the performance of PC based optimization and PLC based optimization can be compared. We can conclude that PLC based optimization is more powerful than PC. Currently, we are comparing the performance of one solver only. The Other solvers can be implemented as part of future work.

Optimization on PLC can also be done using customized code generations for example CVXGEN (online solver for convex programming) can be used deploying the code in to PLC. As embedded optimization has scope in various areas like optimal load distribution in pump, transformer, turbines, generators and compressors. Also we can generate generic libraries which solve general optimization problems. PLC based optimization may not be much accurate but it is time efficient.

More specifically this approach of C-code implementation is more generic and sophisticated. This approach can be used in Real time optimization in terms of energy usage, cost, efficiency, etc.

Finally, we can conclude that optimization algorithms can be put in to PLC and it has potential applications in different areas.

Bibliography

[1]CoinORCOmputationalINfrastructure for Operations Research. Software, 2012. URL http://www.coin-or.org

[2] The Stony Brook Algorithm Repository, Stony brook University, Algorithms in C, http://www.cs.sunysb.edu/~algorith/implement/C.shtml

[3] OOL, Open source optimization library. http://ool.sourceforge.net/

[4] Gurobi Optimization Inc. Gurobi Optimizer. Software, 2012. URL http://www.gurobi.com/welcome.html

[5] IBM. IBM ILOG CPLEX Optimization Studio. Software, 2012. URL: http://www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/

[6] Andrew Makhorin. The GNU Linear Programming Kit (GLPK). GNU Software Foundation, 2000. URL http://www.gnu.org/software/glpk/glpk.html

[7] Hans Mittelmann. Benchmarks for optimization software. http://plato.asu.edu/bench.html , 2011

[8] MIPLIB. Mixed Integer Problem Library. Software, 2012. URL: http://miplib.zib.de/

[9] NETLIB Repository, URL: http://www.netlib.org/

[10] Prof. Dr. Arnold Neumaier, The University of Vienna, Global Optimization http://www.mat.univie.ac.at/~neum/glopt.html

[11] AIMMS,Mathematical Programming Solvers http://www.aimms.com/aimms/solvers/

[12] Solver for quadratic programing qpOASES, URL: http://set.kuleuven.be/optec/Software/qpOASES-OPTEC and qpOASES User's Manual

[13] Dr. Hans Joachim Ferreau, "Model Predictive Control Algorithms for Applications with Millisecond Timescales", URL: http://www.ferreau.eu/science/pdf /dissertation ONLINE.pdf

[14] Dr. Jacob Mattingly,Stanford University, "Code Generation for Embedded Convex Optimization, CVXGEN", URL:http://cvxgen.com/docs/index.html

[15] Application guide ACS355 and AC500 URL: http://www05.abb.com/global/scot/scot201.nsf/veritydisplay/f24a9495dccc3c30c125783 00051d210/$file/en_acs355_ac500_ag_a.pdf

[16] AC500 PLC user Manual URL: http://www.gerrie.com/Content/PDF/abb-ecoplc-aug2011.pdf

[17] Model-Based Design for Automation Systems, Mathworks & Rockwell Automation http://raprod.rockwellautomation.com/resources/downloads/rockwellautomation/pdf/events/automation-fair/2012/tech-sessions/t09_simulation-to-design-test-control-strategies.pdf

[18] Mathworks,Simulink PLC Coder,http://www.mathworks.in/products/sl-plc-coder/

[19] Bart Huyck, Hans Joachim Ferreau, Moritz Diehl, Jos De Brabanter, Jan F. M. Van Impe, Bart De Moor and FilipLogist, "Towards Online Model Predictive Control on a Programmable Logic Controller: Practical Considerations" *Volume 2012, Article ID 912603*

[20] Ivan Rauov´a, Richard Valo, Michal vasnica, MiroslavFikar, "Real-Time Model Predictive Control of a Fan Heater via PLC", *18th International Conference on Process Control, June 14 – 17, 2011 ISBN 978-80-227-3517-9* URL: http://www.kirp.chtf.stuba.sk/pc11

[21] Costas Assimakopoulos, F-N. Pavlidou, "Comparative study of loading algorithms for PLC applications."

[22] BO SVENSSON, "Optimization of Manufacturing Systems Using Time Synchronized Simulation", *Technical Report No R003/2010, ISSN 1403-266X*

[23] D. Kolokotsa, G.S. Stavrakakis, K. Kalaitzakis, D. Agoris. "Genetic algorithms optimized fuzzy controller for the indoor environmental management in buildings implemented using PLC and local operating networks", *Science Direct Engineering Applications of Artificial Intelligence 15 (2002) 417–428*

[24] B. Huyck, F. Logist, J. De Brabanter, J. Van Impe, B. De Moor. "Constrained Model Predictive Control on a Programmable Automation System Exploiting Code Generation: Practical Considerations."

[25] Paolo Dadone and Hugh F. VanLandingham, "PLC Implementation of a Genetic Algorithm for controller Optimization", *Proc. 3rd Int'l Conf. Computational intelligence and Neuroscience (JCIS'98/CI&N'98), 2, 91-94 Raleigh, NC, October 1998*

[26] G. Valencia-Palomo, K. Hilton and J.A. Rossiter, "Predictive control implementation in a PLC using the IEC 1131.3 programming standard."

[27] Bryan T. Griffen, KarelStryczek, RecayiPecen, Teresa J.K. Hall, "A New Approach to Implementing a PLC-Based Model Predictive Controller for Application in Industrial Food Processes."

[28] B. Huyck, L. Callebaut, F. Logist, H. J. Ferreau, M. Diehl, J. De Brabanter, J. Van Impe and B. De Moor, "Implementation and Experimental Validation of Classic MPC on Programmable Logic Controllers"

[29] Vaz, A. Ismael F., and Luís N. Vicente. "A particle swarm pattern search method for bound constrained global optimization." *Journal of Global Optimization* 39.2 (2007): 197-219.

[30] A.I.F. Vaz and L.N. Vicente. PSwarm: "A hybrid solver for linearly constrained global derivative-free optimization, Optimization Methods and Software", *24 (2009), 669-685. (Published version - report).*

[31] Horizon Gitano-Briggs (2010). Small Wind Turbine Power Controllers, Wind Power, S M Muyeen (Ed.), ISBN: 978-953-7619-81-7, InTech, Available from: http://www.intechopen.com/books/wind-power/small-wind-turbinepower-controllers

[32] Real-Time Embedded Convex Optimization Stephen Boyd joint work with Michael Grant, Jacob Mattingly, Yang Wang Electrical Engineering Department, Stanford University

[33]http://www.midaco-solver.comMIDACO website& Solver USER GUIDE

[34]Schlueter M., Egea J.A. and Banga J.R.: *Extended Ant Colony Optimization for non-convex Mixed Integer Nonlinear Programming*, Comput. Oper. Res. 36(7), pp. 2217_2229 (2009)

[35]http://devernay.free.fr/hacks/cminpack/index.html CMINPACK website