Particle Swarm Optimization for ILP Model Based Scheduling

Shilpa KC, C LakshmiNarayana

Abstract—**This paper focus on the optimal solution to the time constraint scheduling problem with the Integer Linear Programming (ILP) modeling for synthesis using Particle Swarm Optimization; an efficient population based search technique.Discrete real space has explored with continuous domain exploration in restricted range and exploitation using nearest discrete value to get convergence faster.The performance of ILP modeling is presented on HLS benchmark and experimental results obtained are promising for NP complete Scheduling problem. This paper presents design complexity is free from complexity of the problem and the experimental results of PSO have proven for a faster rate of convergence to solve constrained scheduling problem**.

Index Terms —Data Flow Graph (DFG), High Level Synthesis (HLS), Integer Linear Programming (ILP), Particle Swarm Optimization (PSO).

I. INTRODUCTION

High-Level Synthesis (HLS) is the process of generating the Register Transfer Level (RTL) design from the Behavioral Description of the digital system [1]-[3]. In HLS, the description of the system is represented in an abstract form usually a Data Flow Graph (DFG), and this representation is transformed and mapped onto architectural elements from a library of resources. The synthesis process involves three major tasks: instruction scheduling, allocation, and binding. Scheduling is the process ofis the process of partitioning the set of arithmetic and logical operations in the DFG into groups so that the operations in the same group can be executed concurrently, while taking into consideration possible trade-offs between the total execution cost and hardware cost. It can be considered the most important step during the architecture synthesis [4]-[6]. Allocation is the process of choosing resources from the library, which involves tradeoffs according to different features like delay, area, power and leakage. Resource allocation is the process of determining the number of functional units of each type for performing operations, memory units (registers) for storing data values and interconnects for data transportation.

II. RELATED WORK

The simplest scheduling technique [1] is AsSoon As Possible *(ASAP)* where the operations in Control and Data Flow Graph (CDFG) are scheduled into control steps from the first control step to the last. An operation is scheduled to the next control step only if all its predecessors have been scheduled. If there is no resource limitation the method generates the minimal number of control steps, otherwise a schedule which requires too many hardware resources to implement might occur. In [2] have given method to avoid the problems of ASAP, technique proposed is the List Scheduling, in this Scheme, ready operations are kept in an ordered list according to a heuristic priority function and are scheduled in order into the next control step until the number of scheduled operations exceed the number of the resources. List Scheduling [3] are faster, but may produce unsatisfactory result because of greedy characteristic and not flexible.Mathematical approaches [4], [5]formulate the synthesis problem as an Integer Linear Programming (ILP) problem presented. ILP approaches leads to exact solutions rather than approximates ones. Secondly these approaches give optimal solutions, But Computation time associated to ILP usually become too large. Simulated annealing based scheduling generates random modifications in the schedule and accepts or rejects them according to a random rule and parameter that gradually decreases with time has given in [6] a scheduling algorithm proposed in [7] to improves an initial schedule by selecting a set of tuples in each iteration pass. The algorithm has a feature of escaping from local minima, but the algorithm does not guarantee for optimality. Scheduling method by Stepwise expansion in HLS, the total computation time can be much reduced compared to the general ILP method, by reduced number of integer variables which appear in ILP formulation by introducing stepwise expansion [8]. This method find optimal or near optimal solution.

Encouraging results from previous work motivated this research effort that tends to explore the effectiveness of instruction scheduling algorithm using Particle Swarm Optimization approach is presented. The algorithm utilizes a novel approach that employs Particle Swarm Optimization (PSO) for scheduling to solve the time constraint scheduling problem. The proposed method was found to generate better scheduling results in the Hardware Abstraction Layer (HAL) differential equation solver high-level synthesis benchmark.

III. PROBLEM FORMULATED

A. *Time Constraint Schedul*ing

The goal of a time constrained scheduler is to realize the design with minimum possible hardware while meeting the time constraint.Tomodel the scheduling problem in High Level Synthesis, binary variables x_{ij} are introduced. In which $x_{ij} = 1$ only when operation v_i starts at time slot$j$. Also, for each operation$v_i, i = 1,2,\ldots\ldots n$, two variables T_i, D_i are defined. They represent the starting time and the duration of the operation v_i,respectively. Thus, the relations between the starting times and the binary variables x_{ij}'s are

$$T_i = \sum_{\forall j} j.x_{ij},\ i = 1,2,3 \ldots. n(1)$$

Assume that the number of available resource types isn_{res} , thus, for each resource type k, the number of components required is $a_k, k = 1,2,\ldots\ldots n_{res}$, for each operation v_i , the function $F(v_i)$ which returns the resource type that can execute the corresponding operation is defined. Now, the time constraint scheduling problem is

$$\mathrm{Min}\left(\sum_{k=1}^{n_{res}} c_k a_k\right)(2)$$

Such that

$$\sum_j x_{ij} = 1, i = 1,2, \ldots n.(3)$$

$$T_i = \sum_{\forall j} j.x_{ij}\ , i = 1,2, \ldots\ldots n(4)$$

$$T_i \geq T_j + D_j, \forall i, j: \text{there is an arc from } v_j \text{ to } v_i(5)$$

$$TN \geq T_i + D_i - 1, \forall\ i: v_i \text{ have no successor}(6)$$

$$\sum_{\forall F(v_j)=K} \sum_{m=j-D_i+1}^{j} x_{im} \leq a_k, k = 1,2 \ldots n_{res}, \forall j(7)$$

where c_k is the cost of using one component of type k. Constraint (3) states that each operation has only one start-time, where constraint (4) represents the sequencing relation between two operations; if there is an arc from T_j to T_i then operation v_i should not starts before the end of the operation v_j. All operations, especially those have no successor, must finish before certain time TN.This stated in constraint (5).Finally, constraint (6) states that the resource bounds must be met at every schedule time step. In order to model the resource constraint scheduling problem, the objective function has to be changed to Minimize (TN) and the values of $a_k's, k = 1 \ldots\ldots n_{res}$ have to be fixed to the available number of components.

IV. PARTICLESWARMOPTIMIZATION (PSO)

PSO's precursor was a simulator of social behavior that was used to visualize the movement of a birds' flock. Several versions of the simulation model were developed [9],incorporating concepts such as nearest-neighbor velocity matching and acceleration by distance. When it was realized that the simulation could be used as an optimizer, several parameters were omitted, through a trial and error process, resulting in the first simple version of PSO. PSO is similar to EC techniques in that, a population of potential solutions to the problem under consideration is used to probe the search space. The PSO algorithm, is based on visualizing the movement of organisms in a flock of birds, and then modeled the individual's behavior of exploration and exploitation mathematically, in the hope of simulating the search behavior of a swarm seen in nature [10]. The methodology of the original PSO is basically uses "particles" flown over the searching space and memorized the best position encountered. The particle adjusts its value of velocity vector, based on its previous velocity vector and the influence of its local best solution and the groups' best solution, and then moves towards the next position. The momentum of each particle tends to keep it from being trapped at local optima, yet but by each particle considering both its own memory and that of its neighbors, the entire swarm trends to converge on the global solution [11]. The main advantage of the PSO is its algorithmic simplicity, convergences rate is faster, and the advantage of dealing with fewer operators is the reduction of computation and elimination of the process to select the best operator for a given optimization

V. EXPERIMENTAL SETUP &TEST DFG STRUCTURE

In the evaluation process, the benchmark HAL differential equation solver high-level synthesis benchmark is been considered and given in the Fig.1.The Data Flow Graph (DFG) consists of many different operations. Use the DFG to formulate ILP for scheduling the flow graph into four control steps, since the DFG contains four different types of operations (i.e. multiplication unit, addition unit, subtraction unit and comparison unit), there is need of four types of function units from library. C_m, C_a, C_s, and C_c are considered to be the cost of a multiplier unit, an adder unit, a subtraction unit and a comparator unit respectively. N_m, N_a, N_s and N_c be the number of multipliers unit (MU), adder unit (AU), subtraction unit (SU), and comparators unit (CU), needed in the final schedule.The ILP formulation for scheduling the DFG into four control steps. If we assume that $C_m = 2, C_a = C_s = C_c = 1$, the cost function has to be minimized and all inequalities have to satisfy.

```
Read(x,y,u,dx,a)
repeat
x1 = x +dx;
```

```
u1=u-(3*x*u*dx)-(3*y*dy);
y1= y + u *dx;
C=x1<a;
x=x1; u = u1; y=y1;
until C;
Write(y);
```

Algorithm 1: VHDL description for HAL example

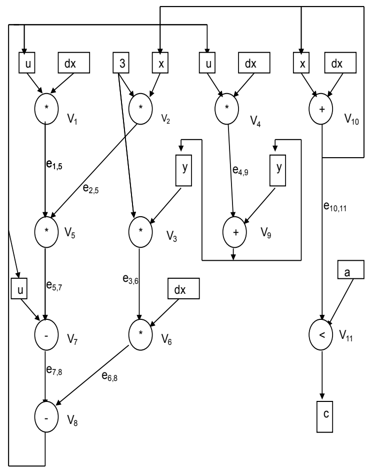

Fig 1. HAL example: DFG of VHDL description

The objective function and ILP formulation is as follow

$$minimize\ C_m \times N_m + C_a \times N_a + C_s \times N_s + C_c \times N_c \quad (8)$$

VI. SIMULATION RESULTS

The objective function (8) is considered, the Population size equal to 200 with Initialization of population as uniform random number in the discrete range of range of [0 1] is taken. Velocity population of PSO is taken as random number in the range of [0 1] uniformly and constant χ is 0.72, c1=c2=2.5, whereas 'w' value decrease from 1.2 to 0.1 for maximum 500 iterations. Penalty factor S is equal to 1000. Algorithms and solutions have developed in the environment of MATLAB.

A fitness function with the use of penalty method has applied to handle the constraints along with objective function as given below

$$F = f + S\left[\sum_{K=1}^{r}\left(g_k^+(x_i)\right)^2 + \sum_{m=1}^{n}\left(h(x_i)\right)^2\right] \quad (9)$$

Where S is penalty factor usually a large value and $g_k^+ = \max(0, g_k)$. g_k and h represent the violation of constraints in (≤ 0) and ($= 0$) cases correspondingly.

A. Steps for Particle Swarm Optimization to solve constraint optimization problem

- PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations.
- In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called pbest. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called gbest. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called lbest.
- After finding the two best values, the particle updates its velocity and positions with following equation

  $$v(n+1) = \chi *[w * v + c1 * rand*(pbest - present) + c2*rand*(gbest - present)] \quad (10)$$

  $$present(n+1) = present(n) + v(n+1) \quad (11)$$

  v is the particle velocity, present is the current particle (solution), pbest and gbest are defined as stated before, rand is a random number between (0, 1), c1, c2 are learning factors, χ is a constriction factor, and w is called inertia weight.
- The process will continue until terminating criteria which is defined as the maximum iterations or minimum error criteria is not attained

Performance shown by PSO is given below. From the graph it is clear that presented concept has delivered the optimal solution within less number of generations.

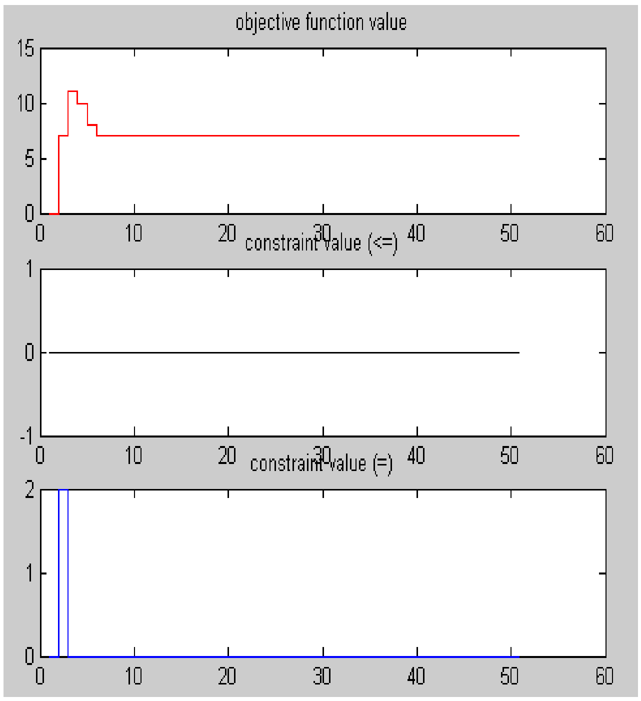

Fig2. Performance of PSO for objective function

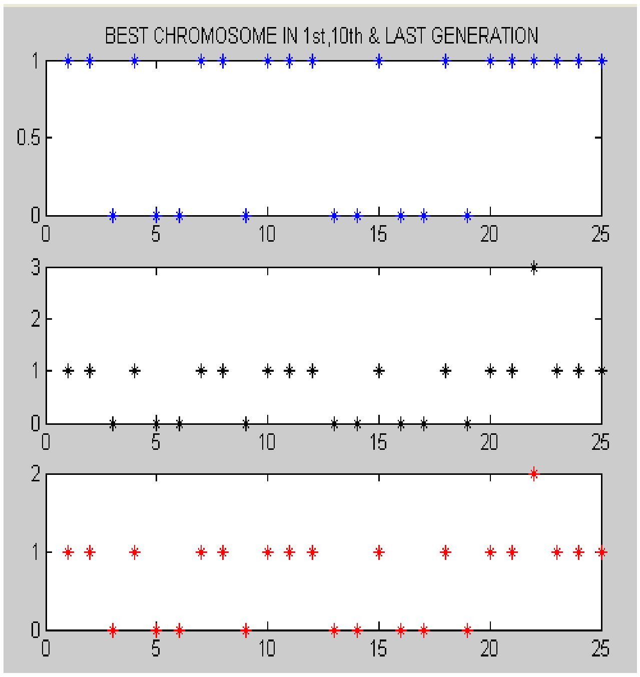

Fig3. Best Solution value distribution in various generation

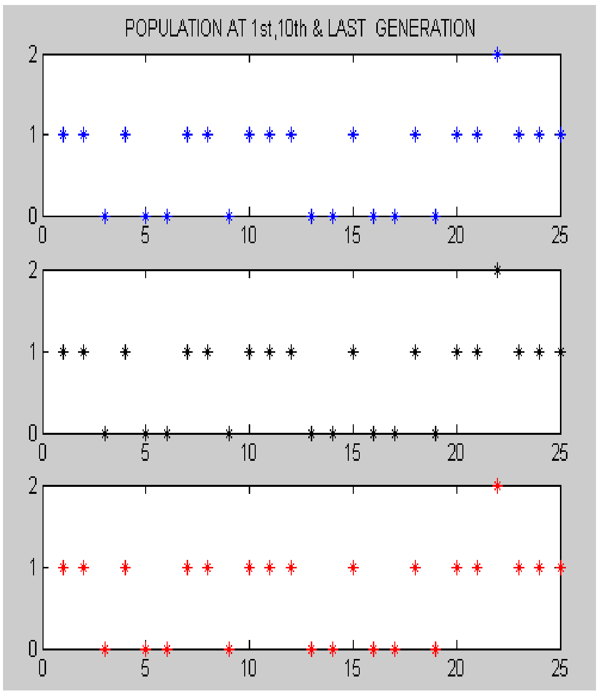

Fig4.Population value distribution with various generation

The final value of resource, scheduling of nodes and the objective function value achieved from Particle Swarm Optimization shown in Fig.5, are as follows:

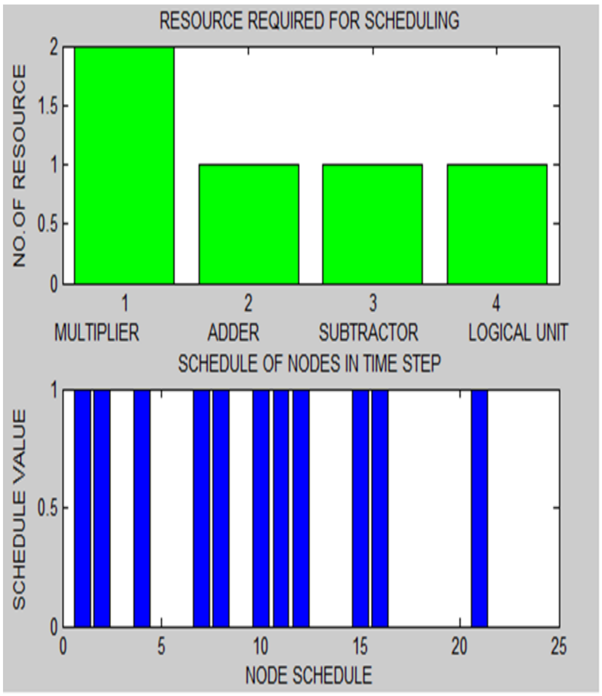

Fig.5. Optimal resource result required for Scheduling

Table 1. Optimal Scheduling of nodes, Resource Requirement

Resource Requirement
$No.of\ multipler(N_m) = 2, No.of\ adder(N_a) = No.of\ subtractor\ (N_s) = No.of\ comparator(N_c) = 1$
Assuming the cost of C_m =2, and $C_a = C_s = C_c$ =1, the minimal objective function values obtained is $2 * 2 + 1 * 1 + 1 * 1 + 1 * 1 = 7$.

VII. Conclusion

We have proposed an approach to the Time constraint scheduling using the concept of PSO. The approach takes the ILP formulation of problem and solved as constraint optimizer. With the proposed method there is very good possibility to explore the search space in better manner. High level synthesis benchmark problem HAL has taken as a test example and scheduling defined for time constraint scheduling. Proposed method is simple and efficient but also design complexity is free from complexity of problem and PSO has shown a faster rate of convergence to solve constrained scheduling problem.

REFERENCES

[1] P. Arato, Z. A,Mann, and A. Orba´n:Time-constrained scheduling of large pipelined datapaths, *inJournal of SystemsArchitecture*, vol. 51, no. 12, pp. 665–687, 2005.

[2] K. Ito, T. Iwata, and H. Kunieda, :An optimal scheduling method for parallel processing system of array architecture,In *Proc. of the Asia and South Pacific Design Automation Conference (ASP-DAC '97),* pp. 447–454, Chiba, Japan, January 1997.

[3] S. O. Memik, R. Kastner, E. Bozorgzadeh, and M. Sarrafzadeh:A scheduling algorithm for optimization and early planning in high-level synthesis, In *ACMTransactions on Design Automation of Electronic Systems,* vol. 10, no. 1, pp. 33–57, 2005.

[4] C.T.Hwang, J.H.Lee, Y.C.Hsu and Y.L.Lin, " A formal approach to the scheduling problem in high level synthesis" . In *IEEE Trans. on Computer-Aided Design*,vol. 10, no.2, pp. 464-475, Feb 1991.

[5] P. G. Paulin and J. P. Knight: Force-directed scheduling in automatic data path synthesis,In *Proc. of the 24th ACM/IEEE Conference on Design Automation*, pp. 195–202,Miami Beach, Fla, USA, June 1987.

[6] P. G. Paulin and J. P. Knight: Force-directed scheduling for the behavioral synthesis of ASIC's,*in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol.8, no. 6, pp. 661–679, 1989.

[7] R. Camposano: Path-based scheduling for synthesis, *in IEEE Transactions on Computer-Aided Design of Integrated Circuitsand Systems*, vol. 10, no. 1, pp. 85–93, 1991.

[8] G. De Michel: Synthesis and Optimization of Digital Circuits,*McGraw-Hill,* Boston, Mass, USA, 1994.

[9] J. Kennedy, and R. C. Eberhart:Particle Swarm Optimization, in*Proc. IEEE Int. Conf. on Neural Networks (Perth Australia), IEEE service Center,* Piscataway, NJ, IV, 1942-1948, 1995.

[10] R. C. Eberhart and Y. Shi: ComparisonBetween Genetic Algorithms and Particle Swarm Optimization, Evolutionary Programming, VII, *Springer Press*, 611-616, 1998.

[11] Maurice Clerc and James Kennedy:The particle swarmexplosion,stability,and convergence in a multidimensional complex space, in *IEEE Trans.on Evolutionary computation* ,vol.6,no.1.February 2002;

Shilpa K.C completedM.Tech degree in General Electronics from Visvesvarya University of Technology (Karnataka, India) in 2007. Now, presently pursing Ph.D in the field of VLSI,Evolutionary Computation from Visvesvaraya University of Technology (Karnataka, India). The object of her interest is in the High level Synthesis, Evolutionary Computation, and Neural Networks.

Dr. LakshmiNaryana.Creceived Ph.D degree from Anna University Chennai. At present he is working as Professor inElectrical & Electronics Engineering Department at BMS college of Engineering, Bangalore. His research areas include Power Electronics, Evolutionary Computation.