New Non-deterministic Approaches for Register Allocation

E. G. Abd-El-Fattah
Computer Department
Mosahmit Beherah Company
Alexandria, Egypt

K. M. Elleithy
Computer Engineering Department
King Fahd University of Petroleum and Minerals
Dhahran 31261, Saudi Arabia
elleithy@dpc.kfupm.edu.sa

Abstract: In this paper two algorithms for register allocation are presented. The first algorithm is a simulated annealing algorithm. The core of the algorithm is the Metropolis procedure. The algorithm presented in the paper has a linear time asymptotic complexity. The second algorithm is a genetic algorithm. The algorithm has a linear time complexity.

Keywords
RISC, register allocation, simulated annealing, genetic algorithms.

I. INTRODUCTION

Most RISC processors make a great effort to optimize register usage to reduce the number of variables that must be kept in memory. The choice of a suitable register allocation technique is vital to the performance of a RISC processor. Register allocation may be viewed as a graph coloring problem, where coloring a graph is an assignment of a color to each of its nodes such that two connected nodes have different colors. Each node in the graph stands for a computed quantity that resides in a machine register. Two nodes are connected by an edge if the quantities are simultaneously live at some point in the object program. The graph coloring problem is an NP-C problem. Exhaustive search approaches for coloring a graph imply an exponential time complexity. Heuristics of reasonable time complexity are required for large size problems. A number of factors can be considered in designing a graph coloring heuristics, e.g., a vertex of high degree is harder to color than a vertex of low degree, vertices with the same neighborhood should be colored alike, and coloring many vertices with the same color is a good idea.

In this paper two non-deterministic algorithms are presented and compared in terms of asymptotic time complexity and quality of solution. These algorithms are simulated annealing and genetic algorithms.

In the last decade Simulated Annealing (SA) has been used extensively in solving various combinatorial problems, e.g., traveling salesman problem [1,2] and quadratic assignment problem [3]. SA is an adaptive heuristic and belongs to the class of non-deterministic algorithms. The heuristic was first introduced by Kirkpatrich, Gelatt and Vecchi in 1983 [2]. The name of Simulated Annealing is inspirited from the process of carefully cooling molten metals in order to obtain a good crystal structure.

A recently proposed paradigm for solving hard optimization problems is Genetic Algorithms (GA). GA has been successfully employed for solving many combinatorial problems [4-7] in areas such as pattern classification, machine learning, scheduling, and VLSI placement and floor planning. Genetic algorithms are search algorithms which emulate the natural process of evolution as a means of processing toward the optimum. A genetic algorithm starts with an initial set of random configurations called a population. Each individual in the population, termed chromosome, is a string of symbols called genes. A solution for the optimization problem is represented by a chromosome. During each iteration, two individuals at a time, called parents, are selected from the population based on a fitness value. A number of genetic operators, as crossover or inversion, are applied on the selected parents to generate new individual solutions called offsprings. These genetic operators combine the features of both parents.

II. SIMULATED ANNEALING ALGORITHM

In this section a simulated annealing algorithm for register allocation is presented. The algorithm is based on the Metropolis procedure. The procedure accepts a new solution with less profit based on a probabilistic function. The objective of the heuristic is to color a graph representing the register allocation problem to maximize a profit function. The profit function is defined using the profit gained from coloring a subset of the nodes. The algorithm presented in the paper has a linear time asymptotic complexity.

The algorithm is shown in Fig. 1. The core of the algorithm is the Metropolis procedure given in Fig. 2. It uses the procedure NEWSOLUTION to generate a local neighbor *new_solution* for any given *solution*. If the profit of the *new_solution* is greater than the profit of the current *solution*, then certainly the *new_solution* is acceptable. If the *new_solution* has less profit, Metropolis will accept the *new_solution* on a probabilistic basis. A random number is generated in the range 0 to 1. If this random number is smaller than $e^{delta/temperature}$, where delta is the difference in profits, then the inferior solution is accepted. The objective of the heuristic is to color a graph representing the allocation problem to maximize a profit function. The profit function is defined as follows:

$$\Pr ofit = \sum_{i=1}^{all-colored-nodes} netsave(node_i) - \sum_{i=1}^{all-uncolored-nodes} netsave(node_i)$$

```
{alpha is the cooling rate < 1};
{beta is a constant > 1};

begin
temperature = initial_temperature;
solution = initial_solution;
time = 0;
repeat
call Metropolis (solution, temperature, cooling_schedule);
time = time + cooling_schedule;
temperature = alpha * temperature;
cooling_schedule = beta * cooling_schedule;
until (time >= Maxtime);
end {of Simulated Annealing}.
```

Fig. 1 Procedure for Simulated Annealing Algorithm.

$netsave(node_i)$: A function representing the profit gained from allocating a variable to a register.

The heuristic proceeds as follows:

(1) Find the set of largest profitable nodes. The number of nodes in this set is equal to the number of available colors.
(2) Color a new node from the uncolored set if it is possible. If there is no more nodes to be colored or time >= maxtime then stop.
(3) Calculate the change in profit after adding the colored node from step 2.
(4) If the change is positive then add the node of step 2 to the colored set, otherwise the node is colored based on a probabilistic comparison.
(5) Go to step 2.

```
Procedure Metropolis (solution, temperature, cooling_schedule);

begin
repeat
get NEWSOLUTION;
delta = new_profit - old_profit
If ((delta > 0) or (random < $e^{delta/temperature}$)) then
solution = new_solution;

cooling_schedule = cooling_schedule - 1;
until (cooling_schedule = 0);
end {of Metropolis}.
```

Fig. 2 The Metropolis Procedure.

Experimental Results

In this section we apply the simulated annealing algorithm for register allocation on a graph of size 128. The graph used for this study is of random nature and random values for the nodes' netsave. The effect of different parameters on the proposed algorithm performance is evaluated. The effect of the following parameters is considered : number of colors, cooling-schedule, beta, temperature, and alpha.

Number of Colors
To evaluate the effect of number of colors on the profit gained from coloring the graph and the number of trials to get the best solution, the number of colors has been changed from 10 to 20. The values for the parameters cooling-schedule, beta, temperature, and alpha are 1.0, 1.0, 100, and 0.8 respectively. The following observations are concluded :

(1) The best solution achieved is 2965. This value is constant for colors greater than or equal to 14.
(2) While increasing the number of colors (i.e. greater than 14), the profit remains constant but the number of trials decreases. The effect of increasing the number of colors is reflected on decreasing the number of trials.
(3) The graph can be colored using 14 colors only while achieving a netsave of 2965.

Cooling Schedule (M)
To evaluate the effect of cooling schedule on the profit gained from coloring the graph and the number of trials to get the best solution, the cooling schedule has been changed from 1 to 20. The values for the parameters beta, temperature, and alpha are 1.0, 100, and 0.8 respectively. The following is noticed :

1- The best solution is obtained at m = 1, and for m > 1 the solution may be less or equal to the solution at m = 1.
2- If the best solution remains constant while m is 2- increased, the number of trials remains constant as well.

3- If the solution decreases as m increases, the number of trials decreases as well.

4- The profit does not change with m if the number of colors is greater than a certain threshold value (14 in this example).

5- As m increases there is a better chance for a negative solution to be accepted. This implies that the solution may be worse or better than the previous solution. For example if the number of colors is 10 and m is increased from 4 to 5, the profit drops from 2565 to 2415 and the number of trials drops from 107 to 99, and if the number of colors is 13 and m increases from 9 to 10, the profit increases from 2685 to 2865.

Beta

To evaluate the effect of beta on the profit gained from coloring the graph and the number of trials to get the best solution, beta has been changed from 1 to 20.The values for the parameters cooling-schedule, temperature, and alpha are 1.0, 100, and 0.8 respectively. The following is noticed :

1- The best solution is obtained at low values for beta. This value is equal to 1 except for number of colors equal to 12 and 13, the best solution is obtained at beta equal to 3.

If the solution remains constant while beta is increased, the number of trials may decrease slightly (number of colors is 15), or remains constant (number of colors is 20).

3- If the solution decreases as beta increases, the number of trials decreases as well.

4- The profit does not change with beta if the number of colors is greater than a certain threshold value (14 in this example).

5- As beta increases there is a better chance for a negative solution to be accepted. This implies that the solution may be worse or better than the previous solution. For example if the number of colors is 11 and beta is increased from 1 to 2, the profit drops from 2705 to 2555 and the number of trials drops from 109 to 101, but if the number of colors is 12 and beta increases from 2 to 3, the profit increases from 2695 to 2785.

Temperature

To evaluate the effect of temperature on the profit gained from coloring the graph and the number of trials to get the best solution, the temperature has been changed from 1 to 100. The values for the parameters cooling-schedule, beta, and alpha are 1.0, 1.0, and 0.8 respectively The following was noticed :

(1) The best solution is obtained at low values of temperature.

2- (2) If the solution remains constant while temperature is increased, the number of trials remains constant as well.

3- (3) If the solution decreases as temperature increases, the number of trials decreases as well.

(4) The profit does not change with temperature if the number of colors is greater than or equal to a certain threshold value (14 in this example).

(5) As temperature increases there is a better chance for a negative solution to be accepted. This implies that the solution may be worse or better than the previous solution. If the number of colors is equal to 11 and temperature is increased from 10 to 20 the profit increases from 2885 to 2895, but if the temperature is increased from 20 to 30 the profit drops from 2895 to 2835.

Alpha

To evaluate the effect of alpha on the profit gained from coloring the graph and the number of trials to get the best solution, alpha has been changed from 0.1 to 0.99 The values for the parameters cooling-schedule, beta, and temperature are 1.0, 1.0, and 100 respectively.. The following is noticed :

1- The best solution is obtained at low values for alpha. This value is less than or equal to 0.6, except for number of colors equal to 11 the best solution is obtained at alpha in the range 0.2 to 0.6.

2- If the solution remains constant while alpha is increased, the number of trials remains constant as well.

3- If the solution decreases as alpha increases, the number of trials decreases as well.

4- The profit does not change with alpha if the number of colors is greater than or equal to 14, except for the values of alpha in the range .95 to .99 where a significant change is noticed.

5- As alpha decreases there is no chance that a negative solution is accepted. That explains why the solution remains constant for alpha less than or equal to 0.6.

6- The profit achieved is very sensitive for values - 6 of alpha greater than.

The simulated annealing algorithm presented in this paper has a linear time asymptotic complexity. The experimental results of the algorithm show optimal solutions in many of the graphs used for testing. The results show better performance compared with other deterministic and non-deterministic approaches.

III. Genetic Algorithm

In this section we introduce a new genetic algorithm for register allocation. A merge operator is used by the selected parents to generate new individual solutions. The number of steps required to examine all pairs in the population matrix to generate candidate's offspring is n^2 (*n* is the population matrix size). Generating an offspring from the parents needs *m* steps (*m* number of node. The experimental results show optimal solutions in many of the graphs used for testing. An outline of a genetic algorithm is shown in Fig.. 3.The algorithm for register allocation is shown in Fig. 4. The algorithm uses the following parameters:

1- Initial population

An initial population consists of any random valid solutions or it can be generated using a starting procedure. The advantage of using a starting procedure is to start with a good solution that can be improved.

2- Hamming distance

Let A and B be any two individual strings of length N. The hamming distance is defined as the total number of positions where $A_i \neq B_i$.

Example

Let A = [0101100], B = [1011100]. A and B are different in positions 1, 2, and 3, i.e., the hamming distance is 3.

```
Procedure Genetic
{ $N_p$ : population size}
{ $N_g$ : number of generations}
{ $N_o$ : number of offsprings}
{ $P_i$ : Inversion probability}
{Population : population matrix of size $N_p$}
begin
  Generate an initial valid population;
  for j=1 to $N_p$
    evaluate fitness(population[j]);          do
  for i=1 to $N_g$ do
  begin
    for j=1 to $N_o$ do
    begin
      choose parents with probability proportional to fitness value;
      perform crossover to generate offsprings;
      for  k = 1 to $N_p$ do
        apply inversion(population[k]) with probability $P_i$;
      Evaluate fitness(offspring[j]);
    end;
    population $\leftarrow$ select(population,offspring, $N_p$)
  end;
  Return highest scoring configuration in population;
end.
```

Fig. 3 Procedure for Genetic Algorithm.

```
Procedure Register-Allocation-by-Genetic Algorithm
{maxtime : is the total time allowed for the genetic process}
begin
  get the population-matrix and the corresponding profit vector ;
  for x = 1 to maxtime do
  begin
    Evaluate the hamming-distance between two individuals;
    If hamming-distance > 1 then
      Merge the two individuals to generate the new offspring;
    If offspring-profit > max(parents' profit) then
      Accept the offspring as a new individual in the population-matrix by deleting any of the parents and replacing the generated offspring;
  end;
end.
```

Fig. 4 Procedure for Register Allocation by Genetic Algorithm.

3- Merge procedure

Let A and B be any two individual strings of length N. An offspring C is generated by merging as follows:

$a_i = b_i$ if $c_i = a_i$

$a_i \neq b_i$, $a_i \geq 0$ and $netsave(a_i) \geq 0$ if $c_i = a_i$

$a_i \neq b_i$, $b_i \geq 0$ and $netsave(b_i) \geq 0$ if $c_i = b_i$

otherwise $c_i = 0$

Fitness value 4-

It is the profit function which is defined as follows :

$$\Pr ofit = \sum_{i=1}^{all-colored-nodes} netsave(node_i) - \sum_{i=1}^{all-uncolored-nodes} netsave(node_i)$$

Analysis

The genetic algorithm presented in this paper may be used to enhance previous results obtained using a starting approach, e.g., simulated annealing. Let n be the population matrix size and m be the vector length (no of nodes). n^2 steps are required to examine all pairs in the population matrix to generate candidate offspring. Generating an offspring from the parents needs m steps. The total number of steps required by the algorithm is n^2m, i.e., the genetic algorithm has a linear time complexity in terms of number of nodes.

IV. CONCLUSIONS

A reduced instruction set computer is a machine with a small number of instructions optimized for a specific application. A great effort is invested to optimize register usage which influences the memory traffic. Two quantities can share a register if their life times are mutually exclusive. The problem of allocating values to registers can be viewed as a graph coloring problem. Each node in the graph represents a computed quantity that resides in a machine register. Two nodes are connected if the residing quantities do not have disjoint life times.

In this paper we introduce a combined two phase approach for graph coloring; a Simulated Annealing heuristic phase and a Genetic algorithm phase. The performance of the Simulated Annealing is controlled by a number of parameters, e.g., number of colors, cooling scheduling, beta, temperature, and alpha. The effect of different parameters on the performance has been thoroughly investigated. A Genetic algorithm is introduced for the second phase. It is used to enhance results obtained from the SA phase. A new efficient genetic operator is introduced to generate an offspring from two parents. The GA proved to enhance the SA results.

Other areas of research related to this combined approach need more exploration. The approach can be extended to include a global phase to measure the effect of applying it across different subroutines. New genetic operators, other than the merging operator introduced in this paper, can be examined and compared with the merging operator.

V. ACKNOWLEDGEMENTS

The first author would like to acknowledge the support of King Fahd University of Petroleum and Minerals for providing the necessary facilities to support this research.

VI. REFERENCES

[1] E. Bonomi and J. L. Lutton, "The N-City Traveling Salesman Problem Statistical Mechanics and the Metropolis Algorithm," SIAM Review, vol. 26, 1984, pp. 551-568.

[2] S.Kirkpatrich, J.C.Gelatt and M.Vecchi, "Optimization by Simulated Annealing," Science, vol. 220, May 1983, pp. 498-516.

[3] R. E. Burkard and F. Rendl, "A Thermodynamically motivated Simulation Procedure for Combinatorial Optimization Problems," European Journal of Operational Research, vol. 17, 1984, pp. 169-174.

[4] J. P. Cohoon, S. U. Hegde, W. N. Martin and D. S. Richards, "Distributed Genetic Algorithms for Floorplan Design Problem," IEEE Trans. Comput. Aided Des., vol. 4, April 1991, pp. 483-492.

[5] L. Davis, "Job Shop Scheduling with Genetic Algorithms," Proceedings of International Conference on Genetic Algorithms and their Applications, 1987, pp. 136-140.

[6] D. E. Goldberg, *Genetic Algorithms in Search Optimization and Machine Intelligence,* Addison-Wesley, 1989.

[7] K. Shahookar and P. Majumder, "A Genetic Approach to Standard Cell Placement using Meta-Genetic Parameter Optimization," IEEE Trans. Comput. Aided Des., vol. 5, May 1990, pp. 500-512.