A Maze Routing Algorithm Based on Two Dimensional Cellular Automata

Shahram Golzari[1] and Mohammad Reza Meybodi[2]

[1] Computer Engineering Department, Hormozgan University, Bandarabbas, Iran
golzari@hormozgan.ac.ir
[2] Computer Engineering Department, Amir Kabir University, Tehran, Iran
mmeybodi@aut.ac.ir

Abstract. This paper propose a maze routing algorithm based on cellular automata. The aim of this algorithm is find the shortest path between the source cell and the target cell , so that the path does not pass from the obstacles. Algorithm has two phases, exploration and retrace. In exploration phase a wave is expanded from source cell and it puts token on cells which it passes via them while expanding. In the retracing phase , we start from target cell, follow the wave and arrive to source cell; the path created in this phase is desirable. Propose algorithm is simple and it's transactions are local and follow the cellular automata properties. This algorithm find the desirable path in $m \times m$ two dimensional CA in $O(m^2)$ time step.

Keywords: cellular automata, routing, maze routing algorithm, physical design, parallel algorithm.

1 Introduction

Maze routing algorithm find the smallest path between the source and target point in a planner rectangular graph. For doing routing by these algorithms, at first entire plan (the place where we want to do routing operation) is simply modeled by grid cells. For doing it , entire plan is displayed by a set of square cells with unit surface which are placed in a two dimensional form just as we have thrown a grid on the plan.

Now, by using these cells and relationship between them, the graph is obtained. In this Manner , each c_i cell is displayed by v_i node in graph. The nodes similar the cells which have obstacle are called closed nodes and other nodes are called non closed ones. If c_i and c_j cells are neighborhood, there will be one edge between v_i and v_j nodes in graph. The weight of each edge in graph equals one, except those edges related to closed nodes, whose weight equals zero.

In the planner rectangular grid graph , each node has four neighbor nodes. The nodes similar to those area of plan which routing can be done by them , are displayed in the non closed form and the another nodes are displayed in the closed form. The aim of maze routing algorithms is to find a path between source and target nodes that this path does not pass any closed nodes and it has also the shortest length. This types of algorithms has two phases: exploration and retrace and algorithm start from

S. El Yacoubi, B. Chopard, and S. Bandini (Eds.): ACRI 2006, LNCS 4173, pp. 564 – 570, 2006.

exploration phase. In this phase we start from source node , consider all possible paths that start from it, and continue all of them so that one of those paths arrive at target nodes.

When path arrives to target nodes, the retrace phase will start. In this phase by using the way of back tracking it is recognized that which nodes of graph settle in the connective path between target and source node. After this phase algorithm is finished. For doing the retrace phase, it is necessary to set some information about paths in exploration phase, then by using this information the retrace phase is done[7,8,11,15]. Maze routing algorithms used in robot path planning and routing phase of VLSI physical design[8,15].

The first algorithm of this type algorithms is introduced by Lee. Which find the shortest path in one grid graph with $h \times w$ dimensions in $O(h \times w)$ [8], then several algorithms based on Lee's algorithm for improving the way of path extension and run time of algorithm, are reported such as algorithms of Soukup[14] and Hadlock[7]. One of the interesting properties of maze routing algorithms is parallelism that hidden on it. This case has caused that these algorithms are simply maps on the parallel structures and in spite of constancy of algorithm complexity, the run time of algorithm, because of doing algorithm by suitable hardware and being reduced the number of instruction done by every processor, are decreased to a large extent. Some of these maps have done by Sagar and Massaka[19].

In this paper, one maze routing algorithm on CA is proposed. The proposed algorithm is so simple. In this algorithm, each cell of CA perform little transactions which cause to reduce the run time of algorithm. The used structure also is simple, parallel and local which is suitable for maze routing algorithm. In this algorithm at first the plan maps on the two dimensional Ca; then by using the simple, local and suitable rules, routing phase is implemented. In this paper we describe the basic concepts of CA in section 2, then explain and implement it by CA in section 3.

2 Cellular Automata(CA)

Suppose a regular network of finite state machine. Each machine called cell. According to a fixed and uniform pattern each cell is related to some of it's adjacent cells. This relationship is local and uniform for all cells. The Cell and all cells related to it are neighbors of cell. Each cell can have one state from limited state at each time step. The states are similar for all cells. At each time step state of all cells update simultaneously and according to uniform rule. This rule is function of states of cell's neighborhood. Therefore in any time next state of cell is dependent on the current state of it's neighborhood. This network start from an initial configuration, at any time step by using the rule for all cells, the configuration is updated and finally the network produce a complex and interest behavior.

According to the explanation the differentiation of CA and other automata network is: simplicity of structure, local communication between cells, establishing a uniform communicative pattern for all cells, simultaneously update of cells, updating cells by uniform rule and producing the complex, interesting behavior from simple cells and rules. According to these properties, CA is referred as a parallel, local and uniform structure in most of literatures, which can simulation the behavior that have these properties[4,6,6,15,18].

Today CA has been used in several areas, such as random pattern generation, computation theory, physical and biological system simulation and applied science[3,5,6,9,12,15,16,18].

So far, by using several method has proved that CA is universal computing model [1,2], but in real world , this model is often used to simulate physical phenomena and this model has drawn attention of researcher of some sciences such as physics and biology rather than that of researcher of computer science. They simulate the evolution of physical phenomena by CA rules step by step. Whereas according the definition , a computational model is not a mechanism for describing the evolution of phenomena; but it is a structure to perform computing on the input data.

In this paper CA is seen as general computing machine and an algorithm is proposed to solve one problem by CA. To solve the computational problems, a data structure and a procedure are needed. The data structure saves input and output data and procedure convert input data to output. The stage of processing and convert input to output in CA done state transmission of CA. Although in the definition of standard CA the memory isn't used but if CA like to play a role of a general computing machine, each automate cell needs several memories to save the values of input and output and also the automata should have the ability to read the values of input and the ability to write the values of output(from-to) memory. The definition proposed follow , which is considered as the definition of CA in this paper, is the extended definition of Mealy Automata[5].

Definition 1: CA is a seven-tuple as $\{Q, d, v, \Sigma, \Delta, \delta, \lambda\}$ when:

1. **Q**: the set of states that each cell can be have.
2. **d**: define the dimension of cell's space. If d=2 , we will have a two dimensional CA.
3. For each cell **x** in CA , vector **V** specify k+1 neighbors that is directly communicate to cell.
4. Σ : the input alphabet of CA.
5. Δ : the output alphabet of CA.
6. δ : the transmission function in the form $\delta : (Q \times \Sigma^n)^{k+1} \rightarrow Q$. According to the function, next state of a cell is dependent on state and values of input memories of all neighbors in the current step. N is the number of input and output registers of cell.
7. λ : the transducer relation which is a finite subset of $(Q \times \Sigma^n)^{k+1} \times \Delta^n$. This transducer defines the value of each output memory according to state and value of input memories of it's neighborhood. Here each cell of CA writes on those memories that reads from them, therefore $\Sigma = \Delta$.

3 A Maze Routing Algorithm Based on CA

3.1 Algorithm Implementation by CA

In this section , the implementation of algorithm by CA is explained. In order to do that, first entire plan should be mapped in a two dimensional CA with von Neumann

neighborhood. In this way, entire plan is considered as e set of cells with have unit area. Those cells that cover the obstacles of the plan are considered with the Block(B) initial state. The initial state of cells that equivalent to the source and target points are considered respectively Source(S) and Target(T) and initial state of other cells are considered Free(F). It is clear that desirable path passes among the cells with F state. Therefore ,the memory variable(state) is needed for save the state of cell. Hereafter we use B-cell, S-cell, T-cell, and F-cell respectively for the cell with state B, cell with state S, cell with state T and cell with state F.

As it described, a wave must be produced from the source cell. For implementation the wave production the below method is used. In the first step, the wave arrives at just cells in the neighborhood of source cell. Now the F-cells that wave arrive to them, update their states to Mark(M). In the next step, wave includes all F-cells which have a M-cell neighbor, that these cells also update their states to M. Expanding the wave continues so and by this way, the exploration phase is implemented. According to this scenario, we can do the wave expanding by simple rule: each F-cell has at least one M-cell or S-cell neighbor update it's state to M.

As it is explained in the idea of algorithm, the wave should put a token in places where it passes, so that the path is made regarding that in retrace phase. While the above scenario does not do this and only expands the wave. Now, the scenario must be verified to save the affect of wave expanding in cells.

As maintained, in each step that the wave expands, number of F-cells change to M-cells, in fact those F-cells that are in the neighborhood of M-cells or S-cell update to M-cells. Then to save the affect of wave, it is sufficient to know that which neighbor of F-cell is M-cell or S-cell and save this information in a memory variable. In order to do this, we use a memory variable called "direction". This variable can save one of the Right, Left, Down up and null values. In initial configuration, the value of this variable is null for all F-cells and T-cell and it is don't care for another cells.

Now, each F-cell which is in the neighborhood M-cells or S-cell, checks that which side cell that neighbor is placed, and regarding that, it assigns value to it's Direction variable. If the M-cell or S-cell is placed at up, down, left and right of cell, the value of Direction variable respectively will update to Up, Down, Right and Left. If cell has more than one M-cell neighbor, it's Direction variable choice one of them randomly. T-cell is also doing operation such as M-cell, but it's state isn't update and only it's Direction variable gets value.

When the wave arrives at T-cell (the Direction variable of this cell gets value that isn't equal to null) exploration phase is finished and retrace phase start. In this phase, the path must be specified from T-cell to S-cell step by step and the cells is placed in the path must be change to P-cells. But how this path is made? To explain this, first we define the following concept:

Definition 2: pointing to: Cell A points to cell B if one of the following cases has been happened:

- Cell A is down neighbor of cell B and Direction value of cell A is equal to Up.
- Cell A is up neighbor of cell B and Direction value of cell A is equal to Down.
- Cell A is left neighbor of cell B and Direction value of cell A is equal to Right.
- Cell A is right neighbor of cell B and Direction value of cell A is equal to Left.

In retrace phase, if T-cell or P-cell has a M-cell neighbor and in addition point to M-cell neighbor, then that neighbor changes to P-cell. The result of using the scenario will be so: when the Direction variable of T-cell gets value against null (that is when wave arrives at T-cell) retrace phase has been started. Therefore , in the first step of new phase, that M-cell which T-cell points to it changes to P-cell; and the next step the M-cell which points to new P-cell changes to P-cell, this operation continues to arrive at S-cell. The desired path involves P-cells. When the path arrives at S-cell the algorithm is finished(That is S-cell has a T-cell or P-cell neighbor).

3.2 Structure of Cells

In the proposed algorithm, each cell has two memory variables called State and Direction. State variable specifies the stat of cell and can get one of the following values:

- Source(S): the source cell has S state.
- Target(T): the target cell has T state.
- Block(B): the cells which the path can't pass through them, have B state.
- Free(F): the cells which the path can pass through them, have F state.
- Mark(M): the cells which the wave arrives at them, have M state.
- Path(P): the cells which placed in the final path, have P state.

Direction variable: this variable specifies the affect of wave expanding in each cell and can get one of the following variables:

- Up: the cells which the wave arrives at them via up neighbor, the value of their Direction variable equals to Up.
- Down: the cells which the wave arrives at them via down neighbor, the value of their Direction variable equals to Down.
- Left: the cells which the wave arrives at them via left neighbor, the value of their Direction variable equals to Left.
- Right: the cells which the wave arrives at them via right neighbor, the value of their Direction variable equals to Right.
- Null: the cells which the wave doesn't arrive at them' the value of their Direction variable equals to Null.

3.3 Initial Configuration of CA

In the initial configuration, the plan must be mapped on CA. To do it, the cells which have obstacles, are being B-cell. The cells similar to source and target respectively get S and T state and state of others cells will be F. The value of Direction variable for F-cells and T-cell equals null in initial configuration and for other cells will be don't care.

3.4 Final Configuration of CA

At the end of algorithm, the cells which place on the path have P state (p-cells). The cells which state of them equal T, B or S in the initial configuration, remain fix.

3.5 Rules of CA

- If State variable of cell equals S, it remains fix in the next step.
- If State variable of cell equals B, it remains fix in the next step.
- If State variable of cell equals F, one of the following cases is done:
  - If cell dos not have M-cell or S-cell between neighbors, it remains fix in the next step.
  - If cell has only one M-cell or S-cell neighbor, the State variable of cell gets M value and cell points to suitable neighbor(if the M-cell or S-cell neighbor has been placed at up, down, left and right of the cell, Direction variable of cell gets Up, Down, Left and Right value respectively.)
  - If cell has some M-cell or S-cell neighbors, the State variable of cell gets M value and cell points to one of them randomly.
- If State variable of cell equals T, the cell follows the rule of F-cell but State variable of cell remains fix.
- If State variable of cell equals M, one of the following cases is done:
  - If cell dos not have P-cell or T-cell between neighbors, it remains fix in the next step.
  - If cell has at least one P-cell or T-cell neighbors and cell points to one of them, the State variable of cell gets P value.
- If State variable of cell equals P, it remains fix in the next step.

3.6 Complexity of Algorithm

The path from source to target in a $m \times m$ two dimensional CA, can't have a length more than m^2, because the path can't pass each cell more than one time and the number of cells is also m^2. Therefore, time complexity of algorithm is $O(m^2)$.

4 Conclusion

In this paper a maze routing algorithm based on two dimensional cellular automata was proposed. This algorithm find a smallest path from source cell to target cell and path doesn't pass the obstacles. The proposed algorithm is simple and has local transactions that match with properties of cellular automata. Each cell of CA also has simple structure and accesses only the contents of neighbor cells in each time and VLSI circuit of it can be designed easily. This algorithm find the desirable path in $m \times m$ two dimensional CA in $O(m^2)$ time step.

References

1. Bruks, W.: Essay on Cellular Automata. Urbana. IL:University of Illinois Press (1970)
2. Conway, J.H., Berlekamp, E., Guy, R.: Wining Ways for Your Mathematics Plays. Vol. 2. Academic Press (1982)
3. Culik, K., Hurd, L., Yu, S.: Computation Theoritic Aspects of Cellular Automata. Physica D. Vol. 45. (1990) 357-378
4. Farmer, D., Toffoli, T., Wolfram, S. (eds.): Cellular Automata Proceedings of An Interdisciplinary Workshop. Amsterdam. North Holland (1984)
5. Gordillo, L., Lunna, V.: Parallel Sort on Linear Array of Cellular Automata. IEEE Transaction on Computers (1994) 1904-1910
6. Gutowitz, A.H.: Cellular Automata. Cambridge. MA:MIT Press (1990)
7. Hadlock, F.O.: A Shortest Path Algorithm for Grid Graph. Networks (1997)
8. Lee, C.Y.: An Algorithm for Path Connection and it's Application. IRE Transaction on Electronic Computers (1961)
9. Mitchel, M.: Computation in Cellular Automata: A Selected Review. Technical Report. Santa Fe Institute. Santa Fe. New Mexico (1996)
10. Packard , N.: Two Dimensional Cellular Automata. Journal of Statistical Physics. Vol. 30. (1985) 901-942.
11. Pan, Y., Hsu, Y.C., Kubitz, W.J.: A Path Selection Global Router. Proceedings of Design Automation Conference (1987)
12. Sarkar, P.: Brief History of Cellular Automata. ACM Computing Surveys. Vol. 32. No. 1. (2000)
13. Sherwani, N.A.: Algorithm for VLSI Physical Design Automation. Western Michigan University. Kluwer Academic Publishers (1993)
14. Soukup, J.: Fast Maze Router. Proceedings of 15th Design Automation Conference (1987) 100-102
15. Toffoli, T., Margolus, N.: Cellular Automata Machines: A New Environment for Modeling. Cambridge. MA:MIT Press (1987)
16. Wolfram, S.: Statistical Mechanics of Cellular Automata. Review of Modern Physics. Vol. 55. (1983) 601-644
17. Wolfram , S.: Computation Theory of Cellular Automata. Communication in Mathematical Physics. Vol. 96. (1984) 15-57
18. Wolfram, S.: Theory and Application of Cellular Automata. Singapor: World scientific (1986)
19. Sagar, V.K., Masara, R.E.: General Purpose Parallel Hardware Approach to the Routing Problems of VLSI Layout. IEEE Proceedings G. Vol. 140. Issue. 4. (1993) 294-304