An Implementation of SAT-Based Two-Terminal Path Finding Using Z3 Solver

YiHong Gu
Tsinghua University
Beijing, China
gyh15@mails.tsinghua.edu.cn

1 Introduction

The 'Two-Terminal Path Finding Problem' has the following description: given $N \times N$ routing grids, and M pairs of terminals, you should use program to find routing paths connecting each pair of terminals. The limit is that different paths cannot cross each other, and no path can cross obstacles. The objective is to maxmize the number of connected pairs, and when all pairs of terminals can be connected, you should minimize the total length of all the paths.

In order to make the objective more clear and not change the original idea of the problem, we simply make the following stipulates that if one grid is defined to be one terminal of pair d, then other pairs $d'(d \neq d')$ can not route across the grid.

2 Related Work

[1] gives a general routing methodology which mainly uses 3 kinds of variable as parameters and use Z3 solver to solve the problem. The abstract idea is that giving a concrete illustration of each pairs: for each pair, define a droplet that starts from the origin and ends in terminal during time t^* and $t^\dagger$, and then define the binary variable $c^t_{p,d}$ to illustrate whether the droplet d is at the position p in time t. It sets a lot of limits to ensure that (1) each droplet d appears in the board during time t^*_d to time $t^\dagger_d$ and it is at its origin position p^*_d in time t^*_d and at its terminal position $p^\dagger_d$ in time $t^\dagger_d$ (2) every second each droplet only appears at one position (3) the droplet satisfy the routing rule that can only step to one adjacent position per second.

Z3 solver [2], developed by Microsoft Research, can use optimized algorithm to check whether it can satisfy all constraints and give a concrete solution. Moreover, it can also maximize/minimize one single variable or expression under all constraints.

3 SAT-Based Model

This part is divided into three main parts, sperately illustrate three different models: the original model which resembles the model established in [1], the model which resembles network flow, the model which combined the 2 models above.

In the beginning, we just emphasize the definitions:

- N: the size of the routing boards.
- M: number of pairs.
- d: the index of one single droplet.
- t: the time.
- p: one single position, can be represented by $p = (x, y) \in \{1, \cdots, N\} \times \{1, \cdots, N\}$.
- P: the set of all p.
- $B(p)$: the set of positions that are adjacent to the position p in four direction(up, down, left, right).
- p^*_d, $p^\dagger_d$: the origin and terminal of droplet d.

3.1 Model 1: Based on the Paper

We use the notation that resembles the original notation in [1], and define the binary variable $c^t_{p,d}$ to illustrate whether the droplet d is at the position p in time t, and the constraints can be divided into four main parts:

to be or not to be: we can simplify the t^* and $t^\dagger$ and assume every droplet appears at the board in time 1 (if we want to connect the two-terminal pair), so the constraints can be (for each d):

$$c^1_{p^*_d,d} \rightarrow \bigvee_{t=1}^{T} c^t_{p^*_d,d}$$

It means: if it appears, then it must reach the terminal.

second conflict: it means every second each droplet only appears at one position, so the constraints can be (for each d):

$$\bigwedge_{t=1}^{T} \bigwedge_{p \in P} \bigwedge_{p' \in P, p' \neq p} \neg c^t_{p,d} \vee \neg c^t_{p',d}$$

position conflict firstly every grid can only be visited once (for each p)

$$\bigwedge_{d=1}^{M} \bigwedge_{d'=d+1}^{M} \bigwedge_{t=1}^{T} \bigwedge_{t'=t+1}^{T} \neg c^t_{p,d} \vee \neg c^{t'}_{p,d'}$$

$$\bigwedge_{d=1}^{M}\bigwedge_{t=1}^{T}\bigwedge_{t'=t+1}^{T}\neg c_{p,d}^{t}\vee\neg c_{p,d}^{t'}$$

Also, for each position p which has obstacles for droplet d, for all t, $c_{p,d}^{t}$ must be FALSE.

travel constraints: it means the droplet satisfy the routing rule that can only step to one adjacent position per second. So for each droplet d, if it appears at position p in time t, it must appears at the adjacent position p' in time $t-1$ (except $t=1$).

$$c_{p,d}^{t}\rightarrow(\bigvee_{p'inB(p)}c_{p',d}^{t-1})$$

optimization object: if we want to maxmize the total pairs, we just need to maxmize

$$\sum_{d=1}^{M}[c_{p_d^*,d}^{1}]$$

If we want to minmize the total length, we firstly enforce that every $c_{p_d^*,d}^{1}$ should be TRUE and then minimize

$$\sum_{d}\sum_{p}\sum_{t}[c_{p,d}^{t}]$$

Note that the notation

$$[statement]=\begin{cases}1, & statement=TRUE\\ 0, & statement=FALSE\end{cases}$$

We can find the scale of the variables and the constraints are very large, we have totally $T\times N^2\times M^2$ variables and almost $O(T^2N^4M^2)$ contraints, which T represents the maximum time used and it can be $\frac{N^2}{2}$ under the worst condition. If we simply make N be 10, M be 4, we can't afford to work it out by our personal computer.

3.2 Model 2: Based on the Network Flow

In this model, we directly use the model resembles network flow architecture, for each droplet d, we build a network flow graph and set the source to be the origin and sink to be the terminal, and all the positions must satisfy the flow conservation. Then we have M layers network flow graph, afterwards we implement the constraint that every position can be only visited once using a constraint like linear constraint.

In this model, if we use some artifice, we can only have totally $2\times N^2\times M$ variables and $O(\times N^2\times M)$ contraints. Besides, after a large quantity of attempts we find if we conbine several contraints together, we can accelerate the speed of our algorithm, and this things can be concretely described in the next section.

3.3 Model 3: The Mixed One

Combine model 1 and model 2 together, we can find a more efficient model, which only have $N^2\times M$ variables and is 1.5 times faster than model 2.

We use the binary variable $c_{p,d}$ to represent whether the droplet d visit across the position p, therefore the contraints are as follows:

limit terms if it is a barrier or is not in the corresponding layer, we enforce $c_{p,d}$ to be FALSE, also we force that every position can only be visited once by the following contraint (for each p)

$$\sum_{d=1}^{M}[c_{p,d}]\leq 1$$

source and sink terms: for each d, we come up with a contraint that

$$c_{p_d^*,d}\leftrightarrow c_{p_d^\dagger,d}$$

connection terms: for each p which is not the terminal/origin/obstacle grid, it must satisfy the following contraint for each d:

$$c_{p,d}\rightarrow(\sum_{p'\in B(p)}c_{p',d}=2)$$

and for each p which is equal to p_d^* or $p_d^\dagger$ for a specific d, it must satisfy the following contraint:

$$c_{p,d}\rightarrow(\sum_{p'\in B(p)}c_{p',d}=1)$$

optimization object: if we want to maxmize the total pairs, we just need to maxmize

$$\sum_{d=1}^{M}[c_{p_d^*,d}]$$

If we want to minmize the total length, we firstly enforce that every $c_{p_d^*,d}$ should be TRUE and then minimize

$$\sum_{d}\sum_{p}[c_{p,d}]$$

4 Optimization

However, the models above don't have efficiency when the graph is sparse. For example, for a board that $N=10$, if we set $M=9$ and place the 9 pairs over the whole board, it runs quite fast (about 1.5 second, model 2), but if we set $M=2$ and place the 2 pairs like the following Figure1, it takes almost 30 minutes to get the best answer.

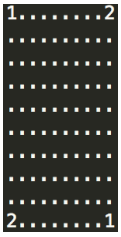

Figure 1: An Extreme Example

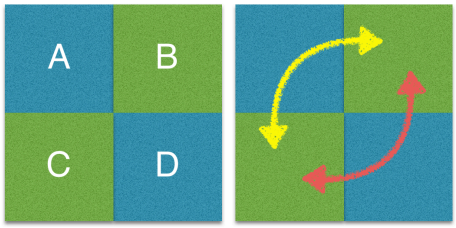

Figure 2: A 4-Block View

In order to optimize the algorithm, we come up with three main points:

(1) We can combine some muti-SAT terms together to a mixed term, which can reduce the check time (it can approximately accelerate the speed twice).

(2) It has no meaning if one droplet d takes the squared area, it means the situation

$$c_{p_1,d} = c_{p_2,d} = c_{p_3,d} = c_{p_4,d} = TRUE$$

should be forbidden where $p_1 = (x, y)$, $p_2 = (x+1, y)$, $p_2 = (x, y+1)$, $p_4 = (x+1, y+1)$, Z3 solver will search all the posibilities including this situation. But, under the circumstances of this specific problem, this situation don't contribute any to the final answer. Therefore, we can manually add the **prune term 1** for each p and d (where $p_1 = p$):

$$\neg(c_{p_1,d} \wedge c_{p_2,d} \wedge c_{p_3,d} \wedge c_{p_4,d})$$

(3) when consider the 4 totally empty grid (no obstacle/terminal), for one single droplet, there are 2 ways to pass from the left-down to up-right, just like the Figure 2.

we can find the path C-A-B and C-D-B are both ok, and Z3 solver will check all the 2 path, so we can forbid this situation simply by the following contraint **prune term 2**(we forbid C-D-B for any p and d):

$$(\sum_{x=1}^{M} [c_{p_1,x}] = 0) \rightarrow \neg(c_{p_2,d} \wedge c_{p_3,d} \wedge c_{p_4,d})$$

where $p = p_1 = (x, y)$, $p_2 = (x+1, y)$, $p_2 = (x, y+1)$, $p_4 = (x+1, y+1)$. Moreover, we can strength the contraint by the following expression if we check other conditions.

$$(\sum_{x=1}^{M} [c_{p_1,x}] = 0) \rightarrow \neg(c_{p_2,d} \wedge c_{p_3,d})$$

and another symmetric terms should also be added (to forbid A-C-D from 2 paths A-C-D and A-B-D)

The prune term 1 and prune term 2 is of great significance, if we use these terms, we can get the answer of example in Figure 1 in 1.5 seconds.

5 Results and Evaluation

6 System Architecture

7 GUI

8 Conclusions

References

[1] Oliver Keszocze, Robert Wille, Krishnendu Chakrabarty, Rolf Drechsler. A General and Exact Routing Methodology for Digital Microfluidic Biochips.

[2] Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In Tools and Algorithms for the Construction and Analy- sis of Systems, pages 337–340. Springer, 2008. Z3 is available at http://z3.codeplex.com/.