Transporting Objects using Cellular Automata Based Modular Robots

Tycho Bismeijer[1] and Stefan Boronea[1]

Vrije Universiteit Amsterdam

Abstract. The problem of transporting objects using modular robotics is relatively new and has seen little interest from the scientific community. However, recent advances in robotics allow small modular robots to cooperate in achieving different tasks that would normally require a more complex approach. In this paper we would like to present a new approach to this problem by designing a cellular automata based model that could successfully be later implemented into the physical world. Our main interest is in finding the appropriate rules that would provide the efficient and realistic support such an implementation needs. Furthermore this study provides an insight into a number of factors that influence the general performance of the system, such as the weight of the objects that need to be transported, the terrain configuration and the number of independent bots that have to perform the task.

1 Introduction

The domain of cellular automata has quickly expanded over the last few decades, from a more general mathematical framework [4] to very specific applications within traffic optimization [1,3], artificial life [5] and physics. These real applications confer a maturity to the domain and provide a stable paradigm for even more extensive applications. One of these applications is modular robotics.

Our goal within this paper is to show the applicability of the cellular automata model in transporting objects within a two dimensional environment. In this sense we have designed a cellular automata that can perform this task. The physical realism of the model was not be considered for the current model, since our goal is to show that a self-organizing system can achieve such behavior. In a nutshell we would like to describe how a cellular automata model can be used to transport objects over gaps within a world in which gravity affects the movements of objects. Our study considers variations of variables such as gap size, object weight and number of cellular automata bots initially placed within the system, which have a direct effect on how efficient the object is transported to the final location.

An important factor for an actual implementation of such a system of modular robots would be power consumption and time consumed for activating a robot to perform a moving or pushing action. Therefore we have considered this as a key point in our simulations, by taking into account the number of reconfigurations performed within the system rather than the time in which the system achieves its goal of transportation.

2 Literature

Our work is inspired by the cellular automata model by Butler et al.[2]. The idea of using a cellular automata to control self configurable robots come from Butler et al., they move bots around a flat surface. This model is then also implemented in different real robots. We extend upon this by considering pushing around objects, and gravity causing bots to fall down.

As mentioned previously, the literature related to cellular automata has been rapidly growing in the past decades. However, the limited interest in practical applications had given the domain a rather exclusive touch until only recently. Among the more significant work done in the modular robotics is Yun's paper [6] that distinguishes two algorithms for modular robot manipulation within a given environment. Nevertheless, the authors do not use a cellular automata model for the reconfiguration of single bot nor do they take into consideration the practical task of transporting an object with their selected model.

3 Model

We model very simple robots like in [2], we call them bots. Furthermore there are obstacles, which provide the bots a surface to stand on, and boxes, which can be pushed around. The bots, as we call them, are able to move around a surface, climb up each other and small obstacles, and to push boxes around. The goal of the bots is to push a box over a surface with a gap. They can do this by filling the gap with bots to supply a surface to push the box over.

This is modelled in a 2 dimensional cellular automata. There is gravity applied to boxes and bots in the downward direction and bots push boxes to the right, as specified in section 3.1. The movement of the bots is determined by cellular automata rules, which are specified in section 3.2.

3.1 The world

Boxes and bots fall down immediately if they are not supported by a box, bot or obstacle under it. The world is initialised with a floor of obstacles on the bottom, to prevent bots from falling out of the world. On top of the first layer of obstacles is another layer with an single optional gap in it with an integer length between 1 and 10.

If a bot has a box to it's right it immediately and automatically pushes thay box to the right, if possible. A box has a integer weight between 1 and 10. Bots can work together pushing a box if they are in a line left of a box. If the length of this line is equal or longer than the combined weight of a line of boxes lying to the right of the bots the boxes get pushed one square to the right. Bots never push a box in a gap, and also never push boxes that have something on top of them.

3.2 Cellular Automata

The bots move according to cellular automata rules, which are presented in figure 1. Bots get randomly selected to activate. There is particular order the bots activate in, and a bot can not activate at all in a time span while another bot gets activated multiple times. If a bot activates it tries to apply a cellular automate rule. The rules are applied in order, the first one that matches gets executed.

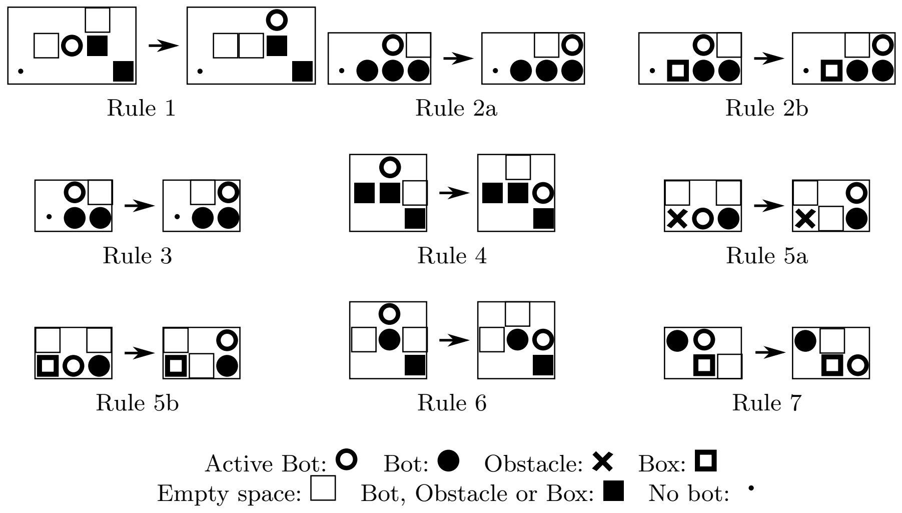
Rule 1
Rule 2a
Rule 2b
Rule 3
Rule 4
Rule 5a
Rule 5b
Rule 6
Rule 7
Active Bot: ○ Bot: ● Obstacle: ✕ Box: ▣
Empty space: □ Bot, Obstacle or Box: ■ No bot: ·

Fig. 1: Cellular automata rules for moving bots.

4 Implementation

The model and application we are studying are new to the literature and therefore our implementation of the model has also been built from scratch. For our simulations we have build a NetLogo application that allows the user to simulate the outcome of the previously described model. The interface also allows the creation of custom worlds and the simulation (in single run or batch mode) of different scenarios in which all or part of the four primary variables are modulated. A typical world view is presented in fig. 2.

Each of the entities involved (i.e. bots, objects and obstacles) are defined as agents within NetLogo. The objects have individual weights, the bots are considered to all have a pushing power of one.

At each time step, the operations in listing 1.1 are performedy by the system. First, a bot is activated randomly and is allowed to perform a movement action

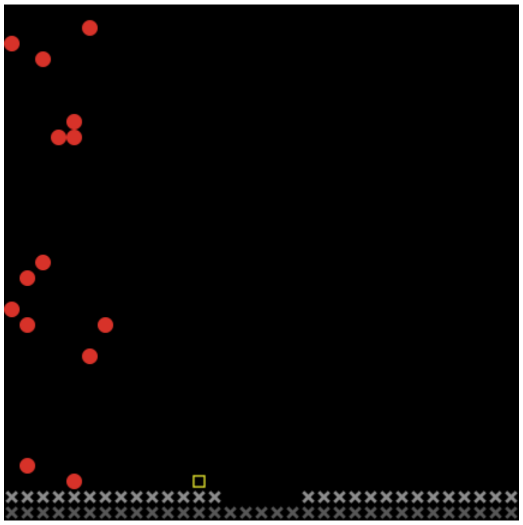

Fig. 2: An initial setting of the world in which the simulation takes place. The red dots denote randomly placed bots, the X marked places show obstacles and the yellow box is the object that needs to be transported to the other end of the world.

based on the implemented rules described previously in section 3. Then all the bots are allowed to make a pushing action, where only the bots that have a box in the immediate vicinity are activated. These proceed to the pushing phase described in listing 1.2. Here the bot finds what the pushing power required for the next obstacle(s) and depending on how many bots can help push the object, it may decide to do so or wait for a new time to act (when perhaps it will be supported by the required number of bots). During the entire process both the bots and the boxes are affected by gravity. This activity is performed at each time step and moves the bot to the lowest possible location to the ground that has the same x coordinate within the world.

Listing 1.1: The operations performed at each time step.

```
to make-single-run
  ; move bots
  ask one-of bots [
    bot-move
  ]

  ; apply gravity and push boxes
  ask bots [
    fall

    ; check if there are boxes to be moved
    push-box

    ; remove boxes at the storage area
    store-boxes
  ]

  ; apply gravity on boxes
  ask boxes [
    fall
  ]

  ;move to next time step
  tick
end
```

Listing 1.2: The actions performed when pushing a box.

```
to push-box
  ; check if there are any boxes to push
  if (any? boxes-at 1 0) ; a box next to the bot
  [
    if(not any? boxes-at 1 1) ;no boxes on top
    [
      let effective-weight [ compute-effective-weight ] of
          one-of boxes-at 1 0
      let pushing-power compute-pushing-power

      if effective-weight <= pushing-power
      [
        ask one-of boxes-at 1 0 [
          move-forward
        ]
      ]
    ]
  ]
end
```

5 Experiments

The experiments we will be presenting in the following sections take into account each of the hypotheses we previously mentioned. For this we have chosen four fundamental variables to study: the gap size, the number of bots used, the weight of a single box and the number of reconfigurations required to solve a specific setting. Out of these, the first three were considered to be independent variables, while the latter was the dependent variable.

We are primarily focusing on determining the following:

1. whether the number of extra robots needed to push over a gap is independent of the gap size;
2. whether the number of reconfigurations is linearly dependent on the gap size;
3. is there a linear correlation between the weight of the object that is transported and the minimum number of bots?

The following experiments were performed in the same world, with the characteristics presented in table 1. We can see that here the box is placed each time before the gap and the bots are placed behind it at random positions (on both the x and y axis).

Table 1: The characteristics of the simulation world.

Variable	**Type**	**Interval**
height	other	32
length	other	32

5.1 Experiment 1

In this experiment we determine how the size of thethe gap determines the minimum number of bots needed to cross over the gap. The minimum number of bots needed to perform a task with certain paramters can be determined from the ratio of succesfull to unsuccesfull runs with these runs with these run. In this experiment there is one box with a weight of one. The number of bots varies from 1 to 20.

The graph of runs with a gap size of 0, 3, 4 and 10 can be seen in figure 3, the graphs the other size look similar. We see that the number of bots needed increases with the gap size up to a gap size of 4. From a gap size of 4 onwards the number of bots needed remains 6. In the simulation we see the group of bots moving over the floor of the gap, without trying to fill the whole gap.

Table 2: The variables involved in experiment 1.

Variable	**Type**	**Interval**
gap size	independent	$[0, 10]$
box weight	independent	1
number of bots	independent	$[1, 20]$
ratio of succesfull runs	dependent	
runs per parameter set	other	100

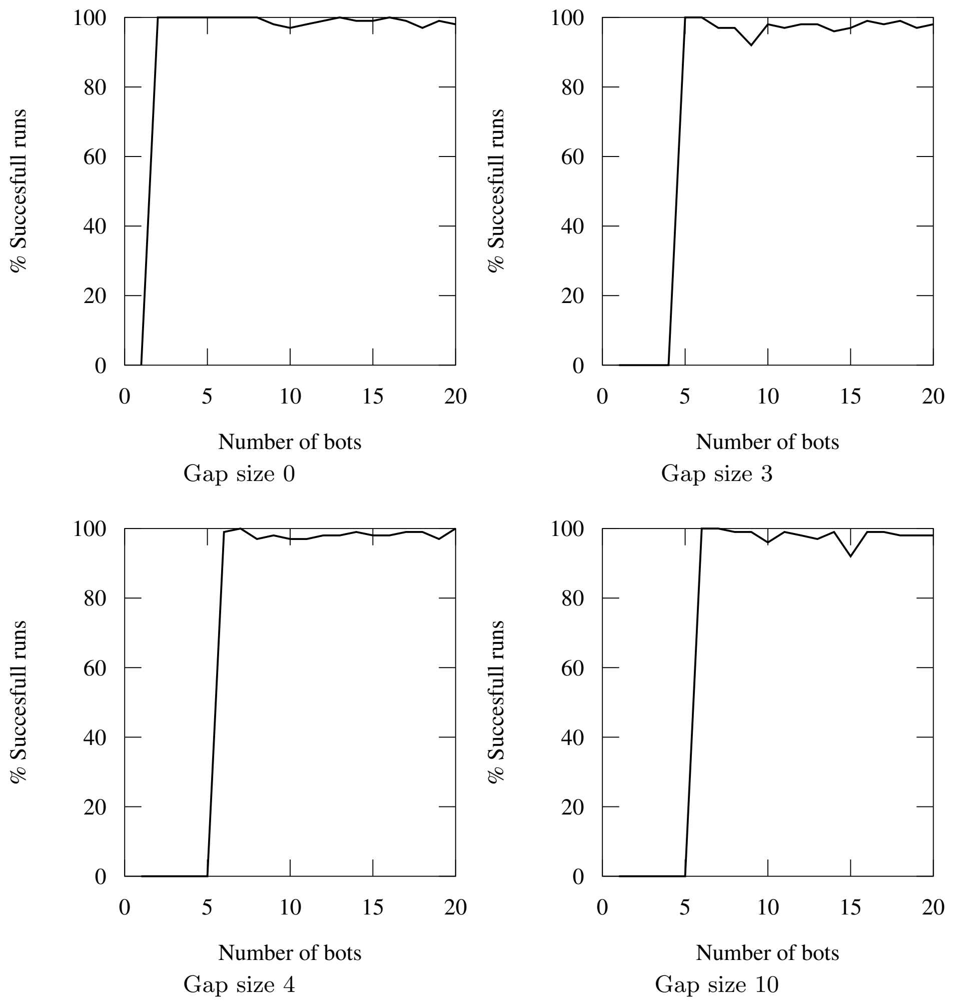

Fig. 3: Succesrate and number of bots for different gap sizes

5.2 Experiment 2

In this experiment we want to determine wether the number of reconfigurations needed to succesfully complete the pushing of the box. The number of bots is kept at 10, we see from the graphs in section 5.1 that this is a representative number of bots. We then measure the number of reconfigurations, which is the number of times a bot activates and performs an action.

In figure 4 we see that the number of reconfigurations is constant under different gap sizes. It doesn't take the bots a significant number of extra reconfigurations to cross gap.

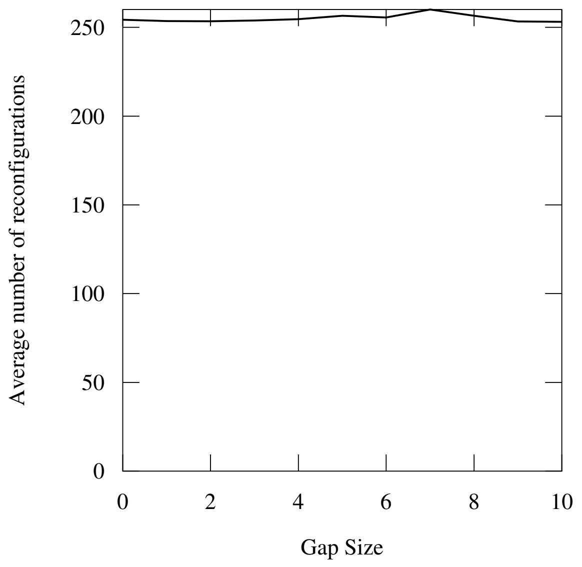

Fig. 4: The number of reconfigurations to the gap size

Table 3: The variables involved in experiment 2.

Variable	Type	Interval
gap size	independent	0–10
box weight	independent	1
number of bots	independent	10
number of reconfigurations		
runs per parameter set	other	100

5.3 Experiment 3

In light of the implementation details provided in section 4, this experiment seeks to determine whether there is a linear correlation between the weight of the box and the minimum number of bots required for the task. The settings for this experiment are shown in table 4.

Table 4: The variables involved in experiment 3.

Variable	Type	Interval
gap size	independent	5
box weight	independent	$[1, 50]$
number of bots	independent	$[4, 50]$ with step 5
minimum no. of bots	dependent	
runs per parameter set	other	10

Fig. 5 shows that the number of required bots to move an object of variable weight. Since the gap size is not actually important for this current experiment, it was set to 5 for all the runs concerning it. We can easily see that this minimum number of bots required increased with the weight of the object linearly, independent of the gap size (the averaged results show the same linear dependency, as shown in fig. 5). This is an expected outcome that validates our initial hypothesis since a heavier object would require a higher pushing power that can only be achieved in our model using multiple bots aligned to create the required force. A different approach to the problem would be to modify a bot's pushing power, thus leading to a more rapid resolve using less bots. However, we will not consider this possibility during our paper since we have chosen to overlook the physical realism of the individual bots (such is the case in [6]).

6 Conclusions

The previous sections have shown how a cellular automata model could be used in the task of transporting a certain object within a gravity-affected world. Also, we have seen how variation in gap size, object weight and number of bots available will eventually influence the outcome of the system. In some cases we studied the interaction effects between these parameters, whereas in other cases we were interested in the overall number of reconfigurations as a definitive measure for the efficiency of the automata rules.

The current study focuses primarily on solving the somewhat simple task of transporting a single box over a single gap of height 1. However, perhaps the most important feature of our model is that its rules could be used in far more complex situations within various world configurations. For example, the current model can be used to transport a convoy of boxes aligned one after

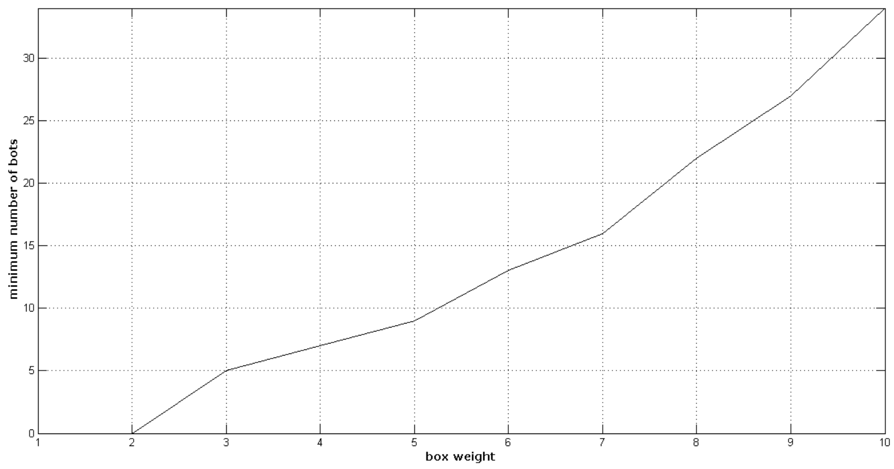

Fig. 5: Minimum number of bots required to push a box of a certain weight.

another, where multiple bots combine their pushing power to move the heavier load. Another interesting extension to the model would be to determine its use in placing certain objects at specific locations within the world so that a certain sequence of tasks is met.

References

1. Bando, M., Hasebe, K., Nakayama, A., Shibata, A., Sugiyama, Y.: Dynamical Model of Traffic Congestion and Numerical-Simulation. Physical Review E 51(2), 1035–1042 (FEB 1995)
2. Butler, Z., Kotay, K., Rus, D., Tomita, K.: Generic decentralized control for a class of self-reconfigurable robots. In: Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on. vol. 1, pp. 809 – 816 vol.1 (2002)
3. Nagel, K., Schreckenberg, M.: A Cellular Automaton Model for Freeway Traffic. Journal de Physique I 2(12), 2221–2229 (DEC 1992)
4. Neumann, J.V.: Theory of Self-Reproducing Automata. University of Illinois Press, Champaign, IL, USA (1966)
5. Nowak, M., May, R.: Evolutionary Games and Spatial Chaos. Nature 359(6398), 826–829 (Oct 29 1992)
6. Yun, S.K., Rus, D.: Optimal self assembly of modular manipulators with active and passive modules. Auton. Robots 31, 183–207 (October 2011), http://dx.doi.org/10.1007/s10514-011-9236-1