

AI for Settlers of Catan

Group 22: Hans Bjerkander, Christian Lidberg, Daniel Mauritzson, Johan Olin

April 20, 2011

Abstract

Settlers of Catan is a very popular board game which the whole project group enjoy. This proposal presents our ideas to develop an AI for a bot, that can compete with other bots, which will be done using a multi-agent system implemented in Erlang. The focus will be on using development cards and to get an AI that not only can win but also get a high average score over all games.

1 Introduction

1.1 What is the problem you are trying to solve?

To create an AI for a bot that can play the board game Settlers of Catan, see rules [4], using a multi-agent system written in Erlang. The focus is to develop an AI that use development cards in a efficient way. The problem can be split into a number of sub problems:

- Choose start positions that fit the strategy.
- How to expand: Upgrading settlements, build new ones or/and get a harbor
- When development cards should be bought.
- Utilizing development cards in a way such that their profit is maximized.

1.2 What results are available in the literature on that problem?

Saleem and Natiq master's thesis [1] is about creating a multi-agent bot for Settlers of Catan. They focus on the trading part of the game and compares how different parameters affect the bot. It uses a static order of how development cards should be played, something we won't use, but still has good description of how to implement the AI. The paper written by Branca and Johansson [2] is also about creating a Settlers of Catan bot and compare it against other bots. In comparison to that paper, this project will try to implement better use of development cards and more strategies regarding the largest army. The paper by Johansson [3] has a good general description on how to implement a multi-agent system[6, ch. 11.4] and how it competes against different AI:s.

1.3 What tools and programs are already available for the problem, or for closely related ones?

There exist a game server written in java, jSettlers2 [7], that is perfect for this problem. It can be modified so that no human player is needed, and it have a set of bots that we can play against. The communication with the server is done through TCP sockets and thus any programming language that supports this can be used to create a bot. This means that less resources is needed for creation of interface and integration with the server, the messages for performing an action and reading the environment already exists. The focus can thus be put in constructing the actual AI.

2 Key ideas and delimitations

2.1 Why is the problem interesting or significant? What will a solution achieve?

Settlers is an interesting game that can be played with many different strategies and many things need to be taken into consideration. A decision taken early will affect the rest of the game. The AI should take care of all the game phases, from picking starting settlements, expanding with new settlements, upgrading to cities, buying/playing development cards and getting largest army/longest road. These phases also change during the timeline of the game, different decisions will change in profit depending on when the decision is taken. There exist implementations that perform relatively well, but they only focus on some of the aspects of the game, e.g. [1, 2]. None of the implementations we have found focused on playing development cards in the most efficient way. So if the project succeeds we will show how to implement another aspect of the game which can be used if one want to combine the existing results to make a complete AI.

2.2 The central idea in your solution

The central idea is to create an AI for Settlers using multi-agents, implemented in Erlang. Erlang is chosen because of its efficient message passing. The agents cover different parts of the game and together they make decisions depending on the environment, which is obtained from the jSettlers2 server. Since the agents are independent it will be easy to change and optimize a single agent without affecting the others. Also the performance will be optimized since the calculations is decentralized to all agents. Though some agents need information from other agents and not only the environment, e.g. the actuator agent that need information from every other agent before it decides the appropriate action. The work distribution in the project group will also be easier with multi-agents.

2.2.1 Agents

Figure 1 shows all the agents and how they communicate.

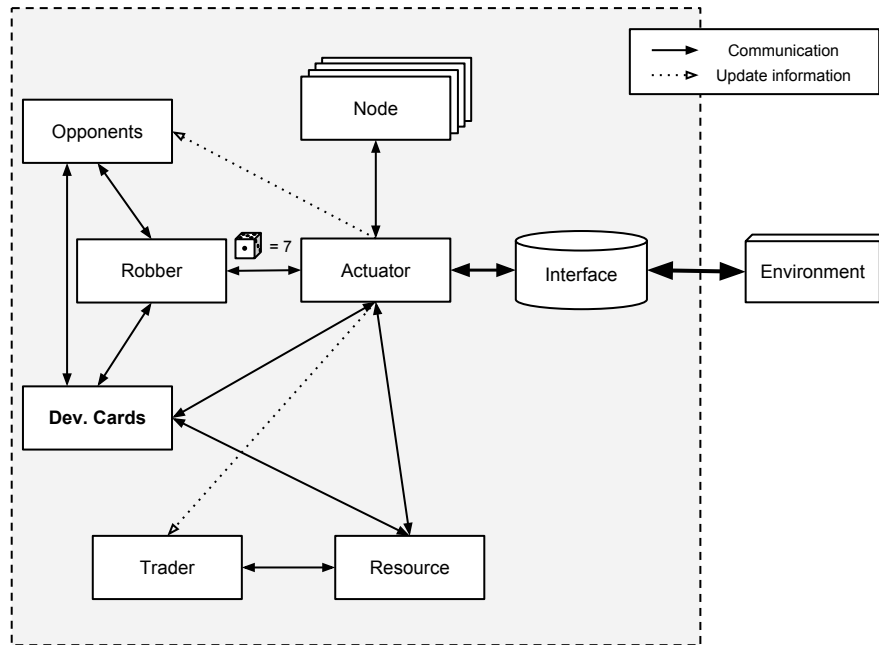


Figure 1: Multi-Agent Architecture

Interface: The agent responsible for the communication with the jSettlers2 server. It forwards the environment and messages from the server to the actuator. The agent also pass the action taken by the actuator to the server.

Node: There exists 54 node agents that keep information of adjacent hexes, which actions that is possible and their utility. The node agents returns if it is possible to build a road, settlement or upgrade to a city. The node agents should also explore its surroundings to calculate the proper utility for e.g. a road built in some direction. It must take in consideration which resources or harbors the AI need and if there is a node in the surroundings that is worth building towards.

Resource: This agent keep track of the resources the AI has and tells the actuator agent about possible buys. It also communicates with the trader agent and include the possible buys, that can be done with trades, to the actuator agent. The resource agent keeps track of how many resources that are in play, which is used by the development cards agent to decide if a monopoly card is worth playing, and what resources the AI need and what

the probability is that the AI will get certain resources. When a seven is rolled and the AI holds more than seven resources it's up to this agent to choose which of the resources to discard.

Development Cards: This agent will be the focus of the project. It will keep track of the development cards held by the AI and, depending on the environment, calculate the profit for playing these cards. It will remember how many and which development cards that have been played and calculate the probability that certain cards will be drawn. Compared with the master thesis [2] where the development cards always is played in a specific order, this project will take many things in consideration.

- The knight cards will be played considering the placement of the robber, the opportunity to get largest army or to defend the largest army and the potential of blocking another player.
- The monopoly card will be played considering the need of certain resources and the opportunity to get many cards of the same resource. It also communicates with the opponents agent to see if this card can destroy another players strategy.
- The year of plenty card will be played considering the need of certain resources.
- The road building card will be played considering the need of roads to build a new settlement and to get the longest road.
- Victory point cards will be held until the game ends.

Robber: The agent that keep track of the placement of the robber and calculates which hex that is best to block if the AI rolls a seven or if we play a knight card. The robber communicates with the actuator, development cards and opponents agent so it can calculate if we can get a needed resource or if we need to block an opponent with a high score.

Actuator: The actuator agent is the one taking the final decision on what action is most appropriate. It passes needed information to all the agents until it receives all possible actions with weights on them and depending on the environment it takes the most preferable action.

Trader: This agent calculates which trades that is possible, though only with bank and harbor due to limitations of the project.

Opponents: The opponents agent keeps track of the opponents resource cards, development cards, victory points, possible moves and so on. With use of this information we are able to make better decisions, e.g. we won't

start building roads towards a potential expansion settlement when there is a high chance that the opponent will build there first.

2.2.2 Strategy

Since we focus on utilizing development cards it's natural to use the card playing strategy [5], which is a strategy where ore and grain is the most important resources when placing the initial settlements. Starting with a high chance to get ore and grain will lead to two fast upgrades and then the resources can be spent on buying development cards. The downside of this strategy is that it often lead to one or two resources are unavailable so it can be hard to expand with another settlement.

Since we won't implement trade with other players we will have to do 4:1 trades with the bank in the beginning. Because of this we need to plan ahead so we know which type of resources we need in the future. Another way to get resources that is unavailable to us is to make good use of the robber, so we need to keep track of which resources each player receives, so we can steal them. In [1] when someone roll a seven and they have more than seven resources they use a round robin selection to choose which resources to discard. This selection strategy is never good since at any point in the game some type of resources is more important than others and especially if some of them are only available with trades. Therefore we will decide which resources we will need in the near future and discard the others. This again means that it's important to plan ahead.

2.2.3 Utilities

Utilities are used to describe how desirable a location is for a certain purpose.

Utilities for the initial settlements: Ore will have the highest utility since three ore are needed for an upgrade and grain will have the second highest. The other resources will have the same utility but we will penalize the AI for each resource that isn't available. Moreover we will penalize the AI more if both wood and clay are unavailable because that will severely lower our chances for an expansion settlement. Harbors will add utility to a placement depending on how useful it can be. E.g. if we have two or three hexes with ore that have high probabilities adjacent to our settlements the ore harbor can be worth starting with.

Utilities for expansion settlements: Resources that are unavailable to the AI will have higher utilities than if they were available to us so we will get a more variety of resources. In this phase the overall utilities for harbors is higher than in the previous phase, but they are still dependent on which

resources we get, because harbors will play an important role when not trading with other players. 2:1 or 3:1 trades instead of the normal 4:1 trades will net more resources and increase the chance of winning. The biggest difference between placing the initial settlements and choosing where to expand to is that the distance from already built roads to the node have to be considered. The further away the expansion node is, the lower the utility will be.

Utilities for building roads: Utilities for building roads will depend on three things:

- The utility of a possible expansion settlement that the road leads to
- The possibility to get the longest road
- The need to secure some location, e.g. if both we and an opponent is competing for a spot we can build one extra road to block the opponent from building on that spot.

2.2.4 Why do you think it will work?

With our literature we can see that multi-agent systems is a good way of implementing an AI for board games [3]. There are also several bots that uses Multi-agent system for Settlers of Catan, with good results [1, 2]. Development cards might not always be the best route to go, but it is a valid strategy that can be combined with the ore/grain strategy[5].

2.3 Define an instance of the problem. How will you measure the performance of your program?

There exist two built in bots to jSettlers2 so we will measure the performance by letting the AI play against them and see how often it win and how many victory points it has on average. We will also keep track of some other statistics to evaluate our AI, for instance what kind of victory points it received and how often it had the largest army or longest road. Also, there exists bots from earlier projects in this course that would be interesting to compete with in the end.

2.4 The scope of your work

This project will be limited in the way that it doesn't handle all phases of the game with the same priority. The placement and expanding of settlements and cities will be implemented but the focus will be on utilizing development cards efficiently and make use of the largest army. To make the most of this we will use the card builder strategy that is described in [5]. Since we will only use one single strategy we will limit us to only play against other bots. A human player would most likely see through our strategy and take

advantage of it. Trading with players will not be implemented, just trading with harbors and the bank.

3 First results

Pseudocode for playing development cards

```
if we got knight card
  if the robber blocks our node/nodes
    get hex values
    if (robber hex value <= max hex value
        OR we can get largest army
        OR we need to defend largest army)
      get resources we need
      send proposal(play knight, place at max hex value,
                    draw from opponent with resource we need)
  else
    get hex values
    if (robber hex value < max hex value
        OR we will steal certain resource we want
        OR we can get largest army
        OR we need to defend largest army)
      get resources we need
      send proposal(play knight, place at max hex value,
                    draw from opponent with resource we need)

if we got monopoly card
  for each resource type, r
    get how many cards we will get when playing monopoly
    generate all possible hands with trading
    send proposal(play monopoly, r, [all possible hands])

if we got year of plenty card
  generate all possible hands we can get with two free resources
  send proposal(play year of plenty, [all possible hands])

if we got road building card
  send proposal(play road building)
```

3.1 Formula for calculating hex value for robber placement

$$HexValue = \sum_{n=1}^6 B_n * R * O_n * D$$

$$B_n = \begin{cases} 2 & \text{if city in node } n \\ 1 & \text{if settlement in node } n \\ 0 & \text{else} \end{cases}$$

$R =$ Weight of resource in hex

$$O_n = \begin{cases} -2 & \text{if we own the city/settlement in node } n \\ 1 & \text{else} \end{cases}$$

$D =$ Probability of the hex

References

- [1]
 - 1. Saleem, H. and Natiq, R.R. (2008). A Multi-agent player for Settlers of Catan , Masters thesis,
[http://www.bth.se/fou/cuppsats.nsf/all/5182dee54d5efc33c1257559004f8dbc/\\$file/SOC_Final_Report.pdf](http://www.bth.se/fou/cuppsats.nsf/all/5182dee54d5efc33c1257559004f8dbc/$file/SOC_Final_Report.pdf)
 - 2. The paper is about developing a Multi-agent bot for Settlers of Catan. The bot is focused on the trading and uses develop cards without any strategy.
 - 3. The most important reference to this paper in our document is in Section 2.1 and 2.2.4

- [2]
 - 1. Branca, L. and Johansson, S.J. (2007). Using Multi-agent System Technologies in Settlers of Catan Bots , Conference Paper,
[http://www.bth.se/fou/forskinf.nsf/alfs/c3607f15e9248a9cc12572eb0053320b/\\$file/BrancaJohanssonABSHLE07CR.pdf](http://www.bth.se/fou/forskinf.nsf/alfs/c3607f15e9248a9cc12572eb0053320b/$file/BrancaJohanssonABSHLE07CR.pdf)
 - 2. This paper is about developing a multi-agent Settlers of Catan bot and comparing it against two other monolithic bots. The papers bot was bad but the paper contains a good description of the implementation of a multi-agent system and what to improve.
 - 3. The most important reference to this paper i our document is in Section 2.1 and 2.2.4

- [3]
 - 1. Johansson, S.J. (2006). On using Multiagent Systems in Playing Board Games , Conference Paper,
<http://portal.acm.org/citation.cfm?id=1160737>
 - 2. This paper is about using a multi-agent bots in boardgames and compares bots in the games Diplomacy and Risk.
 - 3. The most important reference to this paper i our document is in Section 2.2.4

- [4]
 - 1. Klaus Teuber (2007), Settlers of Catan: Game Rules, http://www.catan.com/en/download/?SoC_rv_Rules_091907.pdf
 - 2. The official rule book for Settlers of Catan
 - 3. The most important reference to this paper i our document is in Section 1.1

- [5]
 - 1. Scott MacPherson (1999), Settlers of Catan Strategy and Tactics Guide, Tactics Guide, <http://files.meetup.com/1550090/Settlers%20of%20Catan%20Strategy%20and%20Tactics%20Guide.pdf>

-
2. A thorough guide on different strategies and phases the game goes through.
 3. The most important reference to this paper i our document is in Section 2.2.2 and 2.2.4
- [6]
1. Stuart Russell and Peter Norvig (2010), Artificial Intelligence A Modern Approach, Pearson Education
 2. The course book
 3. The most important reference to this paper i our document is in Section 1.2
- [7]
1. Jeremy D. Monin ,jSettlers2, webpage, <http://sourceforge.net/projects/jsettlers2/>
 2. The projectpage of jSettlers2
 3. The most important reference to this paper i our document is in Section 1.3