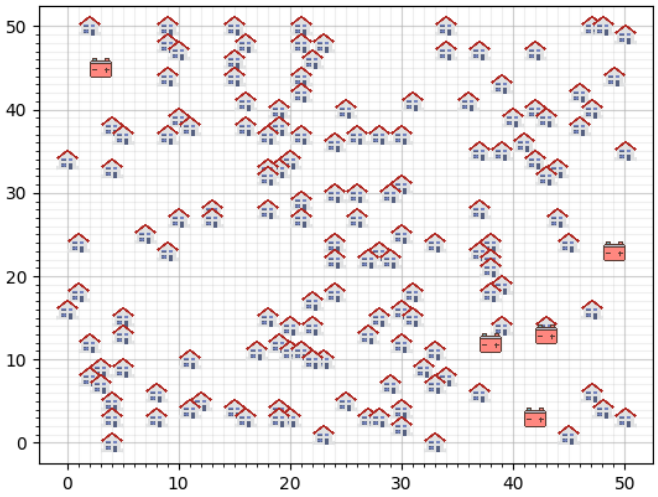
Smart Grids

A case for Heuristic Education



Abstract:

This paper attempts to design a case for an introductory heuristic computer science class. The concept of smart grids is used to teach students about effective ways to deal with NP-hard problems. The case is set up asking the students to find a cost wise most optimal configuration and placement of batteries, receiving excessive power from solar panels of local households. To make it more manageable for students it is split up in five subproblems. In this paper, the hardness of such a problem are discussed as well as the parametrization of case specific order parameters. In this case these are the output of houses and the cost of the wires towards the batteries. By the end a full potential heuristics case is laid out.

30th June 2017
Jasper Bakker (10260250)
Stijn Verdenius (10470654)

UNIVERSITY OF AMSTERDAM

1. Introduction

Renewable energy generation, such as wind and solar, is growing fast around the world (Beaudin, Zareipour, Schellenberglabe & Rosehart, 2010). However, as Beaudin et al. (2010) mention, these energy sources have highly variably energy generation which can cause technical and economical challenges when used on large scale. They state that energy storage is widely considered to be a potential solution to this problem. As explained in a report by the International Energy Agency (2011), one of the characteristics of a smart grid is the accommodation of local, sustainable, energy generation and storage.

In this paper a fictional case is introduced for such a local smart grid. The smart grid case is a set of combinatorial problems written for students with an introductory level computer science background. In general, the students receive a cartesian grid with coordinates for houses with a solar panel on them and the task to connect them to batteries to store the excessive productions of these panels. These batteries are deemed to be connected to the already existing energy net and only need to be connected to the houses with solar panels. The goal is to make sure there is no overcapacity on these batteries, and to minimize the cost of this setup as much as possible, using heuristic algorithms. It is therefore a constraint satisfaction problem.

1.1 The case

In the case each battery has a maximum capacity and the energy output amongst the houses differ. In general students need to assign (e.g. connect) each house to one of the batteries. To make it more manageable for the students the case has been divided in exercise A through E. At first battery location and type is fixed, in later exercises the student can vary both. Both the batteries and the wires cost money. Furthermore, they can only submit valid solutions, that is when all houses are connected and none of the installed batteries are over capacitated.

The objective for exercise A is to get a valid solution. For this exercise, the type and location of the batteries are fixed. This exercise is just practice to get the framework running and for the students to meet the constraints. In exercise B however, they are asked to calculate the cost of their setup of exercise A and improve the cost by swapping a few houses in battery-assignment. For this purpose, a cost function is introduced:

$$Cost = \sum_{i=0}^{B}(B_i * C_i) + (L * W_c)$$

B is the total number of batteries B_i stands for an individual battery, C_i for the cost of that battery, L for the total Manhattan distance wirelength and W_c for the cost per wire segment. Next, in exercise C, things get more difficult. Now the students must move the batteries within the grid whilst simultaneously reassigning houses to batteries to find their most profitable cost. Thereafter, in exercise D, the students receive a list of batteries (see appendix B) of which they can choose their own battery configuration and optimize even further. Finally, in exercise E, they will have to implement new path planning methods because they will now get a steep penalty when a wire runs underneath a house to fully solve the smart grid problem. The full case description, as will be available to the students, can be found in the appendix C.

1.2 What makes a good case

According to lecturer Daan van den Berg a good heuristics case has the following requirements. A case should be easy to read and understand. It should be computationally solvable. It should not be solvable by just using google. It is potentially solvable by iterative as well as by constructive algorithms. It should resemble reality as closely as possible, however, this should not make the case educationally less interesting. It should be easy to start the case. Especially considering the amount of programming or the difficulty of the data structures, the case should not have a steep start. Furthermore, it resembles the other cases (Berg, 2017) algorithmically. Finally, the case should have a twist somewhere, so solving it wouldn't be doing just more of the same, causing students to lose interest.

1.3 Where the hard problems are

To make sure the case has an increasing degree of difficulty, it is important to understand which problems are hard. Within the field of computer science, four (partly overlapping) computational complexity categories are defined. A problem is categorized by its worst case (Hayes, 1997). The easiest problems are the problems for which an algorithm is known that can solve it in polynomial time, these problems fall in the category of P (Hayes, 1997). Harder are problems that fall in NP, for which there is no algorithm known that can find a solution in polynomial time, however, if a solution is given, it can check if it is right within polynomial time (Hayes, 1997). Then there are problems that fall in NP-Hard. Problems are said to be NP-

Hard if they are at least as hard as the hardest NP problems, but they do not necessarily fall in NP (Johnson, 2012). NP-hard problems are also never decision problems but optimization problems. For these problems, the entire state space needs to be considered to make sure the most optimal solution is found. Finally, there are NP-complete problems (Monasson, Zecchina, Kirkpatrick, Selman & Troyansky, 1999). However, these are not important to discuss further in the context of this paper.

Although the computational complexity is characterized by its worst-case scenario, the average case is of interest as well. Hayes (1997) states that for the 3-SAT, a well-known constraint satisfaction problem, there are phase transitions regarding the computational complexity. He explains that the computational cost, is relatively low regarding under constrained problem instances that always result in a solution or over constrained instances that very rarely result in a solution. However, for instances with a solvability in between those over- and under constrained instances, closer to the phase boundary, the computational costs are much higher. Furthermore, he states that it seems well established in the literature that phase transitions in SAT are intrinsic to the problem itself, not a remnant of any algorithm. According to Monasson, Zecchina, Kirkpatrick, Selman and Troyansky (1999), these phase transitions seem to occur among more NP-complete problems, but are not excluded to NP-complete problems.

As mentioned a good case has a gradual increase in hardness with a relatively ease first part. For exercise A, we have the setup for a typical case of a NP constraint satisfaction problem, because the order parameters are chosen within the phase transition. Exercise B is already a NP-Hard problem. Because it is a constraint optimization problem no guarantee can be made that the optimal solution is found without checking the entire state space. The state space increases more than polynomial when for example more houses or batteries would be placed. When students arrive at exercise C and D, the problem is still NP-Hard, but since the state space increases, it becomes increasingly harder from B to D. Finally, for exercise E the state space is equally large as in D, however it is harder to find good solutions. An approximation of the state space for each exercise can be found in the appendix (B).

2. Materials and methods

2.1. Materials

The smart grid case was constructed using Python on computers running a Windows 10 operating system. An overview of the used libraries and the specifications of the computers used can be found in tables A and B respectively in the appendix A. The case description for the students can be found in the appendix C.

2.2. Methods

2.2.1. Grid construction

In this first version of the smart grid case, three grids have been constructed. The dimensions of all three cartesian grids are 50x50. Each grid has 150 houses on them that were randomly positioned on the grid. Furthermore, for solving part A through C, 5 training wheel batteries have also been put at random locations. In the next section their parametrization is being discussed.

2.2.2. Capacity and Deviation

As mentioned before, each house has its individual output to the battery. To make this process not trivially configured and make sure the solvability falls in the phase transition, these houses cannot all have the same output. For this reason, there has been parameterized on the (training wheel) battery capacity and the variation in output.

To do so an attempted solve has been made 100 times on each of the grids, every time with a different output-variation and different battery capacity. The results of these 100 attempts are counted on how many times they could solve the grid concerning constraints. For the variation, the mean output of all 150 houses was set on 50 with a deviation of 5, 15, 25, 35 and 45, then the output of the houses was drawn from a uniform distribution between this upper and lower bound. The total capacity of all the houses combined was normalized to 7500. The capacity per battery was consequently varied from 1500.75 through 1521.75 with increments of 0.75. The cumulative capacity of the five batteries was thus always more than the cumulative capacity of the output of the houses.

The algorithm used in attempt to solve these grids was done with a hillclimber, we will refer to this as SolverA from hereon. What SolverA does is constantly making a small random change in the assignment of houses to batteries and then only keeping that change if it both gets us closer to a solution and decreases the cost. There is two ways it can make these changes. Firstly, it can swap two houses their battery assignment. This is most beneficial if both batteries are not over capacitated or equally over capacitated. Secondly it can reassign a single house to another battery, this on the other hand is a more valuable move

if one battery is more over capacitated than the other. The hillclimber stopped after it didn't make any improvements anymore for 1000 iterations.

After the test, three board configurations were chosen based on solvability of the parameters so that they are in the phase transition. One of where the solvability was about 50%, one where it was about 30% and one where it was about 7%. These boards where the verified with a series of test of 10.000 random walks of 50.000 iterations. They were also attempted to solve by SolverA, but now with 23.000 iterations of no improvements as exit clause, to conclude sufficient hardness but solvability for the students. These output configurations could thereafter be used to test exercise C through E and therewith also parametrize the wire cost and battery cost for exercise D and E.

2.2.3. Wire cost and Battery configuration

After having finished the first three exercises students reach the point where they must choose their own battery configuration. Where before no actual cost of batteries and wires but only the wirelength was of real importance because the amount of batteries was fixed, it now becomes relevant that they will have a way to calculate the cost of their setup. To do so a fixed battery scheme was established (see appendix B) with the largest battery having slightly more capacity than the training wheel batteries. The cost per capacity unit becomes cheaper over the batteries. This would motivate students to choose the most of the big batteries as needed. However, wires cost money as well, and therefore, given the proper wire cost, an interesting equilibrium could appear for the cost wise most optimal battery configuration. If wires would cost next to nothing it is wiser to place big batteries. Yet if wires would cost relatively much, it would be wiser to place a lot of small batteries close to the houses.

In this paper, an attempt is done to parametrize the wire cost in such a way that this equilibrium would appear. The aim is to choose the wire cost in such a way that it is not trivial for students which battery to place.

The algorithm used for this will be referred to as SolverB and efficiently uses SolverA since constraints still need to be met. The process is as follows: first randomly assign all houses to a battery. Then run SolverA, both working towards the constraints and optimizing the costs. Thereafter a K-means algorithm is used, which moves the batteries from their current position towards the centroid of all the houses that are assigned to them. These last two steps are repeated until the batteries stop moving in the second step ergo the setup converges.

SolverB is run 100 times for every possible battery configuration of which the sum of their capacities, using the small and the big battery (appendix B), is between 7500 and 10800 for every wire cost. The results are analysed for the configuration which returns the best cost on each wire cost. Moreover, the ratio big/total batteries is used to find the best candidates for the eventual wire cost.

For these wire cost the same procedure of running SolverB 100 times was followed to check whether the chosen wire cost would result in an interesting equilibrium for three battery types as well. To analyse this, the standard deviation of the battery types contribution to the total capacity is used. If this standard deviation is low, then the capacity is relatively equal distributed among the three batteries.

3. Results

3.1 Output of the houses

The solvability for just meeting the constraints show a phase transition for all three grids, between a capacity per battery of 1501.5 and 1521.75. Figure 1 below shows the results for grid 1. The phase transitions are steeper for higher variations. Furthermore, is the grid easier solvable for higher variations, given a certain capacity per battery.

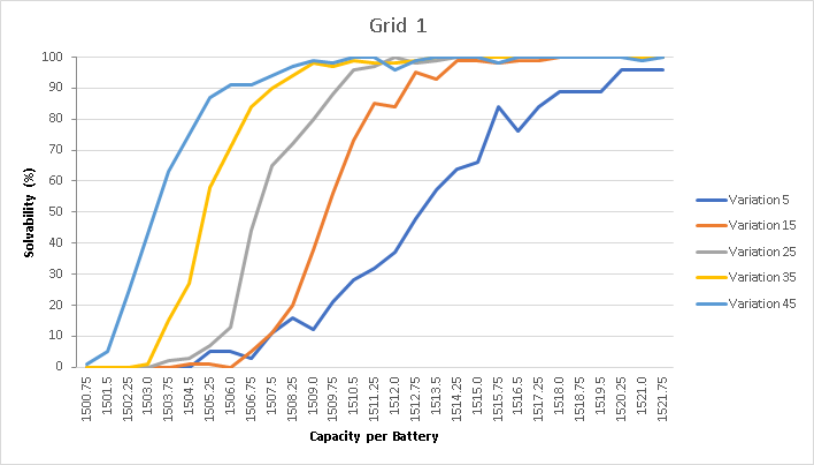


Figure 1: solvability grid 1

The results for the other two grids, showing similar phase transitions, can be found in appendix D. The final instances that were chosen to be in the case, and the results of testing the amount of solutions that can be found for these instances can be seen in table 1.

Grid number	Battery Capacity	Variation	Valid solutions SolverA (%)	Mean iterations SolverA	Mean no. Hillclimber improvements	Valid solutions Random Walk (%)
1	1507	25	59%*	57098.95	135.46	0.061%*
2	1508.25	15	36%*	57517.85	162.99	0.066%*
3	1506.75	5	5%*	47859.81	126.69	0.068%*

*All valid solutions were unique solutions

Table 1: Summary final grids

The percentage of found solutions for all three grids corresponds to the solvability depicted in Figure 1 and Figure A and B in Appendix D. This is the case even though the hillclimber of SolverA for the test of the final boards had a 23 times higher exit clause than for the initial parameterization. Also, it can be seen that it is a lot harder for SolverA to find valid solutions for grid 3 than for 2 and for 1, while the unique solutions for the random walk are found with about the same frequency which might predict a lot of local minima. Finally, the mean amount of iterations and the amount of Hillclimber improvements are not indicative for the amount of solutions found.

3.2 Wire cost

As can be seen in in figure 2 below, a clear correlation can be distinguished between the wire cost and the ratio of big/total batteries. Fitness score is calculated by sorting the mean of the 100 SolverB scores per configuration on how well they scored and then take their place in line as their fitness (on average over the 3 grids). The horizontal axis is a value between one and zero corresponding to the percentage of big batteries (see appendix B) in the configuration. Thus, each scatter point in the graph represents a battery configuration.

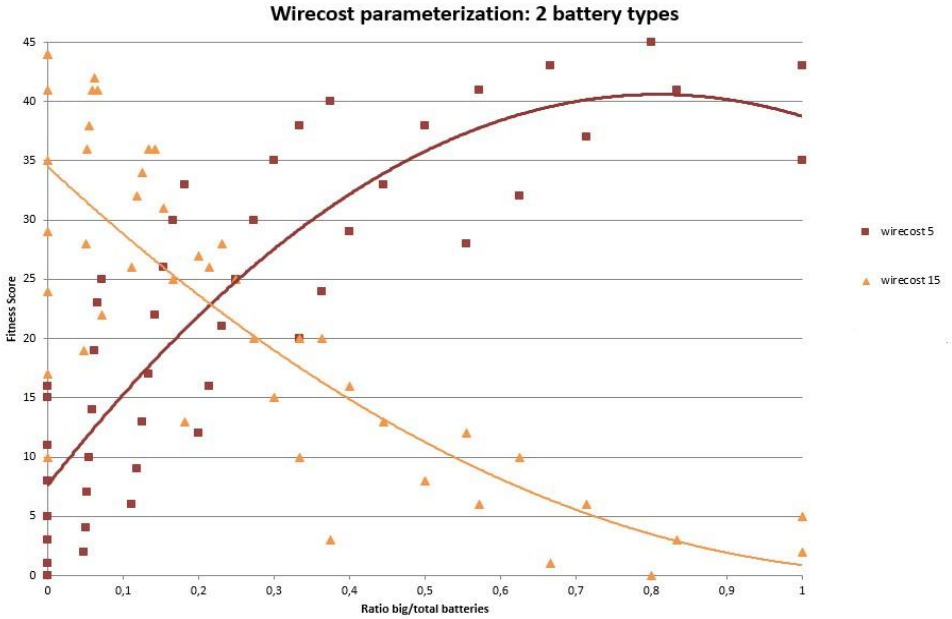


Figure 2: Parameterization of the wire cost for two battery types at wire cost 5 and 15

As can be seen in figure 2, for a relatively low wire cost of 5, a large part of the better scoring configurations have their average solution with a large proportion of big batteries. For the relatively high battery cost this is the opposite. At wire costs laying in between 5 and 15, the best scoring configurations tend to lie more in the middle, having a peak at a certain ratio. The most equal distributions were found for wire cost of 9 and 10. In figure 3 the graphs for wire cost 9 and 10 can be seen. Of all tested wire costs, cost 9 has its peak closest to an equal distribution of number of batteries. Wire cost 10 has its peak closest to an equal contribution of each battery type to the total capacity, which is at about 0.25 small to big batteries.

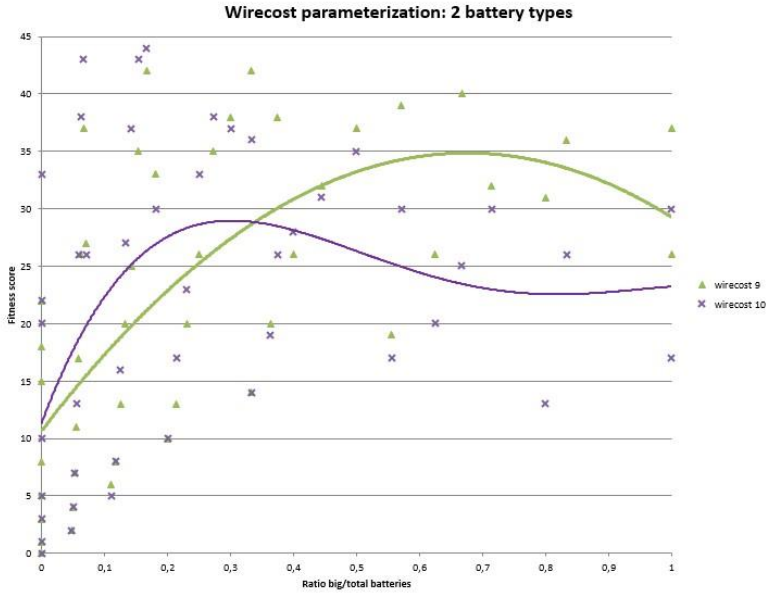


Figure 3: Parameterization of the wire cost for two battery types at wire cost 9 and 10

However, in exercise D and E, three battery types can be chosen by students. In figure 4 the same fitness score is plotted but now on the horizontal axis the standard deviation of the battery types to the capacity distribution is depicted. A higher standard deviation meaning here that the total capacity needed is less equally spread among the battery types. For the case, especially the higher fitness scores are relevant, since we expect the students will improve upon their answer until they have a good score. Therefore a larger amount of high fitness scores with a low standard deviation are considered to be an

indication for a good final wire cost. As can be seen in the graph the configurations resulting in a good fitness score for wire cost 9 or 10 have a lower standard deviation than those for wire cost 11.

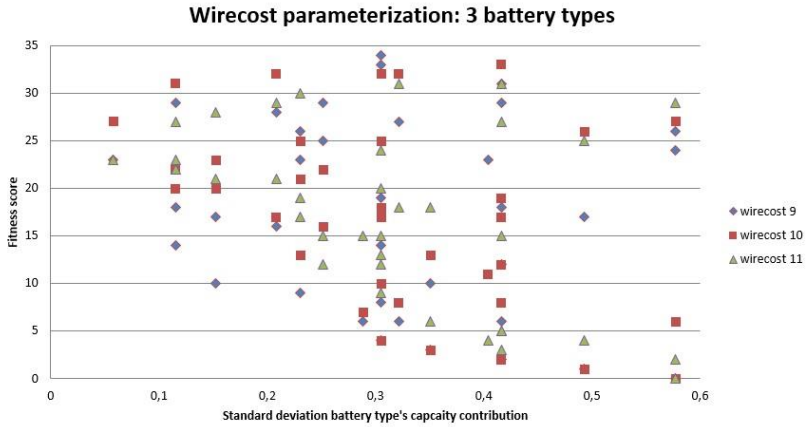


Figure 4: Parameterization of the wire cost for three battery types at wire cost 9, 10 and 11

3.3 Conclusion

As expected, a higher battery capacity leads to easier solvable instances. A higher variation of the output however, also has a positive effect on the solvability. This might be considered counterintuitive because one might expect it to be harder when the different outputs have a higher deviation since there will be more outliers. An explanation for this might be the fact the outputs were drawn from a uniform- instead of a gaussian-distribution. With a higher standard deviation, the uniform distribution gives the algorithm more output combinations to try to fit each battery. Whereas a Gaussian distribution would have most packages still near the mean and just some awkward outlier outputs where the algorithm wouldn't know what to do with. For example, if a battery has 70 capacity left, it would be impossible to fit 2 houses out of the distribution with 5 deviation since they would be between 45 and 55. However, for the distribution with 45 deviation, there are multiple solutions for 2 houses to precisely fit the battery because there would be more small outputs.

Table 1 seems to indicate that the state space of grid 3 is has the most local optima, after that grid 2 and then grid 1. This is plausible because the random walk shows about the same amount unique valid solutions, while the local search algorithm SolverA finds a lot more valid solutions for grid 1 and having a tougher time with grid 3.

Furthermore, a wire cost to optimal configuration correlation is clearly distinguished, although it is clearer to make a statement about them when comparing small to big than when comparing three types of batteries. Nevertheless, wire cost 9 and 10 seem to be favourable for relatively equal battery configurations, even though they are not as ideal as a perfect split in thirds. Specifically it seems that wire cost 9 predicts a more economically optimal configuration of quantity of batteries whilst wire cost 10 does so for the contribution per type in capacity.

4. Discussion

This paper attempts to justify the choices made in creating a new case for heuristics education. However, some questions are left unanswered.

For instance, it was found that a higher capacity per battery and a higher variation of the output both lead to easier solvable cases. The latter being rather surprising. Further research must be done to find out the specific cause.

Moreover, considering the three final grids, further research might use a simulated annealing algorithm to see this mitigates the differences in solvability between the three grids. If this is the case, the state space is confirmed to have various local optima, as was expected by our analyses.

In addition, a limitation of SolverA is that it's hillclimber stops iterating after 1000 iterations without improvement. But for each swap there are 22.950 battery assignments possible. Even though the final boards have been run with an exit clause of 23.000 iterations without improvement, this still doesn't guarantee that all possible states around a certain point in the state space are visited, since the hillclimber is a stochastic algorithm.

Likewise, SolverA does not first search for a valid solution and then for a lower cost. When it does not have a valid solution yet, it always seeks to improve the cost and tries to get closer to a valid solution at the same time. Otherwise it will swap back. Therefore, SolverA might have found less valid solutions then it would otherwise have, making the solvability of the grids look harder than they might be.

Additionally, a good case should be solvable with both iterative as well as constructive algorithms. Although we expect that it could be solved with a depth first algorithm, the case has not been tested for it.

Furthermore, In the earlier stages of creating the case there has been a plan to make it more realistic by using housing blocks instead of single houses, randomly distributed across the grid. This might have given the grids more structure. As mentioned by Fischer, Stützle, Hoos and Merz (2005), a more structured instance makes their Traveling Salesman Problem instances easier. This might also be the case for the smart grid instances Further research must conclude if the same is the case for the smart grid case.

Finally, for the wire cost parameterization we assumed it to be less trivial of the wire cost would be not ó too high and not too lowô However further research must be done to be able to conclude which wire costs are harder to solve for than others. This research only studied a select number of possible configurations with 2 and 3 batteries and moreover it also only used a limited amount of different wire costs, which were all integer values.

5. References

Beaudin, M., Zareipour, H., Schellenberglabe, A., & Rosehart, W. (2010). Energy storage for mitigating the variability of renewable electricity sources: An updated review. *Energy for Sustainable Development*, *14*(4), 302-314.

Berg, D (2017, april 4th). main_page url: www.Heuristieken.nl

Cook, S. A. (1971, May). The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing* (pp. 151-158). ACM.

Fischer, T., Stützle, T., Hoos, H., & Merz, P. (2005). An Analysis Of The Hardness Of TSP Instances For Two High Performance Algorithms. In *Proceedings of the Sixth Metaheuristics International Conference* (pp. 361-367).

Hayes, B. (1997). Computing Science: Can't get no satisfaction. *American scientist*, *85*(2), 108-112.

International Energy Agency. (2011). Smart grids roadmap [Research report]. Retrieved from https://www.iea.org/publications/freepublications/publication/smartgrids_roadmap.pdf
Johnson, D. S. (2012). A brief history of NP-completeness, 1954ï 2012. *Documenta Mathematica*, 359-376.

Monasson, R., Zecchina, R., Kirkpatrick, S., Selman, B., & Troyansky, L. (1999). Determining computational complexity from characteristic ó phase transitionsô *Nature*, *400*(6740), 133-137.

Appendix A

Table A: computer specifications

Laptop manufacturer and type	**RAM**	**Processor**	**Operating system**
Lenovo ideapad 710S-13ISK	8.00 GB	Intel Core i7-6500U CPU @ 2.5GHz, 2592 MHz, 2 cores, 4 logical processing units	Windows 10.0.14292 Build 14393
Hp Elitbeook 8570w	8.00 GB	Intel Core i7-3630qm CPU @ 2.4GHz	Windows 10

Table B. software used during the research

Programming Language:	Python 2.7.13
Libraries:	Random
	Time
	Operator
	Math
	Sys
	pickle
	Matplotlib (version 2.0.1)
	Numpy(1.12.1)
	deepcopy
	pandas

Appendix B

Table A. Battery scheme

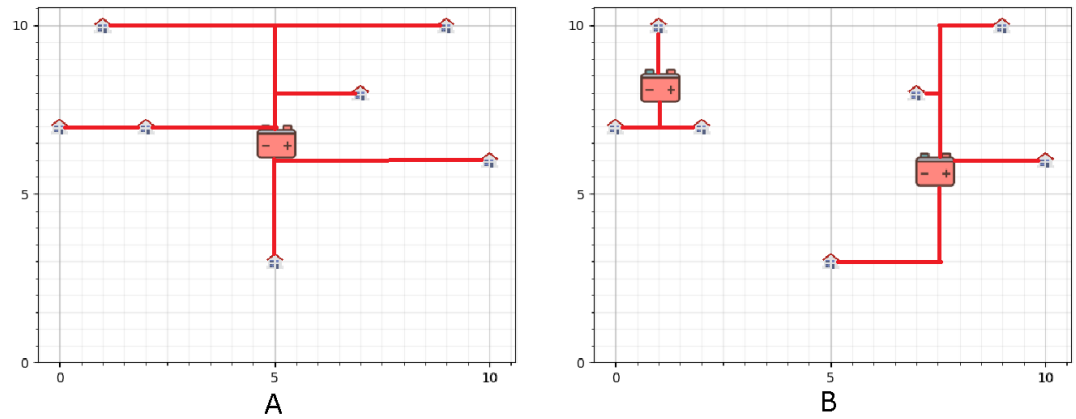
Battery	Capacity	Price
G1	1507	5.000
G2	1508.25	5.000
G3	1506.75	5.000
Small	450	900
Middle	900	1.350
Big	1800	1.800

Table B. State Spaces

Part	Explanation	Approximation
A	150 houses that must be connected to 5 batteries:	150^5
B	Same as A	150^5
C	150 houses that must be connected to 5 batteries of which all the batteries can be moved to 50*50 different locations	$150^5 * (50^2)$
D	150 houses that must be connected to y batteries. An upper boundary would be that every house has its own battery, so y would be 150. There are n amount of battery configurations between a capacity of 7500 and 150^3 a loose estimate of the state space takes the configuration with the largest amount of batteries: y (small) batteries. Again all batteries can be placed on 50*50 places.	$150^{150} * (50^2)$
E	same as D	$150^{150} * (50^2)$

Appendix C

Introduction
Green energy is the energy of the future. Unfortunately, most means of green energy production do not have a constant output, since the sun doesnôt shine at night and the wind is quite fluctuating as well. With more households getting solar panels and turbines, it makes sense to store the energy in batteries at peak hours so it can still be used when production is low. This way the batteries can be used as a constant energy source on the already existing energy-delivering network (the net). The new infrastructure, a *Smart Grid* of consumption-production, is by no means trivially configured. So, the challenge is: ôWhich houses should be connected to which batteries, and where should they be placed?ô



This neighbourhood has seven houses with solar panels. As you can see in setup A the total wire length used for the setup is nearly twice the amount in B because the tactical placement of an extra battery. Do these savings in total wire length amount to the cost of placing an extra battery however?

Assignment
To keep things a bit manageable a few simplifications have been made. The variable time has been taken out of the equation for example and all houses are on a grid. The assignment is to find a setup that gives an optimal cost for wires and batteries. Wire length is measured in *Manhattan distance* and its price is 9 per segment. There is one *hard constraint* however, each battery has a maximum capacity and is not allowed to be over-capacitated output to avoid explosion danger.

A) The city council of the neighbourhoods that opted for a smart grid have proposed a few favourable spots

where the batteries must be placed. Connect all houses (same as the maps above) to one of the batteries. Apart from the feasibility, costs are also an issue. These fixed batteries all cost 5000. The cost function is given by: Cost = sum(battery * battery typeôs price) + (wire length * 9)

B) Calculate the cost of your setup and try to optimize it further (if possible)

It is figured out that the batteries might not have been in the best place to begin with. It is voted to choose new positions for the batteries.

C) Try to improve the results found in exercise B by moving the positions of the batteries.

Due to the research done earlier in A, B and C, SomeBatteryCompany Inc. saw some feasible business opportunities. They made three new types of batteries which are listed below. The residents of the neighbourhoods ask you to come up with a plan to store all their energy as cost-efficiently as possible. You can place batteries anywhere, and use as many of each battery type as you think you need.
Battery types:

#1	Cap: 450	Price: 900
#2	Cap: 900	Price: 1350
#3	Cap: 1800	Price: 1800

D) Configure a new set up for the Smart Grid, for as low as possible cost.

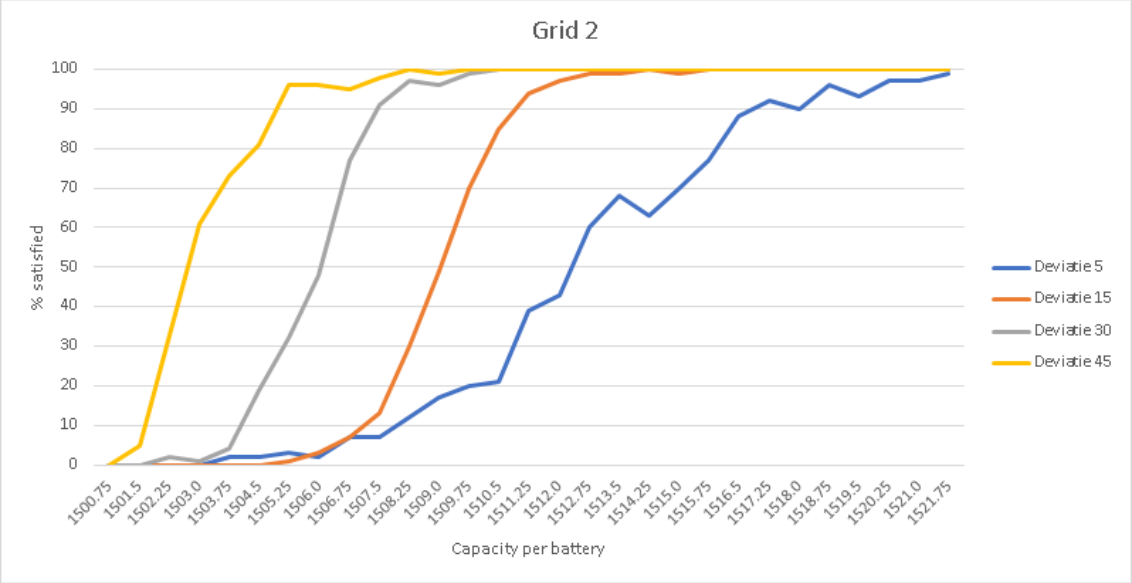
New legislations enact that residents having to tolerate cables run underneath their houses are entitled to a compensation of 10.000 euro. This seriously changes the cost scheme for any configuration.

E) Configure a new set up for the Smart Grid, for as low as possible cost.

Advanced) Create a series of new battery-schemes, when are they harder to solve and when is it harder to choose which battery to place?

Appendix D

Graph A. Parameterization of battery capacity and deviation for grid 2



Graph B. Parameterization of battery capacity and deviation for grid 3