GridPenguin: A District Heating Network Simulator

Wu, J.; Everhardt, Rob; Stepanovic, K.; de Weerdt, M.M.

Publication date
2022

Document Version
Final published version

Published in
Conference Proceedings New Energy for Industry 2022: 2nd Conference of the Innovation Network, October 13-14, 2022 in Linz, Austria

Citation (APA)
Wu, J., Everhardt, R., Stepanovic, K., & de Weerdt, M. M. (2022). GridPenguin: A District Heating Network Simulator. In C. Gradwohl, A. Degold, & T. Kienberger (Eds.), *Conference Proceedings New Energy for Industry 2022: 2nd Conference of the Innovation Network, October 13-14, 2022 in Linz, Austria* (pp. 132-141). NEFI: New Energy for Industry. https://www.nefi.at/files/media/Bilder/News/NEFI%20Konferenz%202022/NEFI2022%20Conference%20Proceedings/NEFI_Conference_2022_Proceedings.pdf

GRIDPENGUIN: A DISTRICT HEATING NETWORK SIMULATOR

Jichen WU[1,*], Rob EVERHARDT [2,*], Ksenija STEPANOVIC[3], Mathijs DE WEERDT[4]

[1] Flex Technologies, Atoomweg 7-9 Utrecht the Netherlands, Jichen@flextechnologies.ai
[2] Flex Technologies, Atoomweg 7-9 Utrecht the Netherlands, Rob@flextechnologies.ai
[3] Delft University of Technology, PO Box 5 Delft the Netherlands, K.Stepanovic@tudelft.nl
[4] Delft University of Technology, PO Box 5 Delft the Netherlands, M.M.deWeerdt@tudelft.nl
* Corresponding author and submitter in ConfTool

Abstract: District heating system (DHS) optimization is becoming an increasingly important problem because of the unused potential in flexibility that could allow less energy being wasted and the integration of renewable energy. While new optimization methods are proposed every year to tackle this problem, the literature lacks a good way to benchmark newly proposed methods. To address this problem, we introduce GridPenguin, an open-source computational simulator for the physics of district heating networks. It provides flexibility in usage by providing building blocks with which the user can build any grid he wants. The detailed simulation of the physical world with a focus on the heat balance and average flow rate and temperature allows for fast and accurate simulation. By explaining the physical equations and computational model as well as the comparison to existing software, we lay a solid foundation for the performance of the simulator. We present GridPenguin as a metric to evaluate optimization methods as well as a tool for easy integration of advanced machine learning methods into DHS optimization. The source code of our project can be found on https://github.com/ftbv/grid-penguin

Keywords: District heating system; Simulation; Optimization; Heat production planning

1 INTRODUCTION

Heat networks can play an important role in the energy transition by reducing the dependency on fossil fuels. Also they can provide flexibility in their use and/or production of electricity, an important feature in the future power system. Currently in many district heating networks, the decisions on heat and power production are often made separately. The decision on heat is made purely based on the experience of the grid operator, and the temperatures are often kept relatively constant throughout a season.

In the literature, various scientific attempts have already been made to empower operators to reduce heat losses and use the flexibility of DHS for electricity balancing using various methods including MILP, NLP and Genetic algorithms [1] [2] [3] [4] [5].

However, to the best of our knowledge, the literature of DHS optimization has two crucial limitations. First, there exists no benchmark to evaluate and compare different papers and methods. A paper proposing a novel method usually evaluates it in the exact same model it uses for optimization, and the proposed method is only compared with simple or traditional strategies instead of other methods from the literature [1] [3] [6]. Second, mathematical optimization methods dominate this field. They require a deep and comprehensive understanding of DHS from the researchers to simplify the complex, non-convex DHS to a solvable linear or convex model. It might be beneficial if less domain knowledge is required and data-driven machine learning methods can be used. However, there is no standard dataset, nor a standard way to generate reliable data. Getting data directly from the grid can

be very difficult and is not flexible enough for collecting data for rare situations. On the other hand, we lack a reliable model to generate artificial data that simulates real grid accurately enough.

To address the above-mentioned limitations, we propose GridPenguin: a comprehensive, accurate and still relatively fast simulator of a DHS. Various modules that model components in a DHS are provided to the user, and these modules are backed-up with equations describing the physics of the heat system and the hydraulic system [7]. After the user specifies the grid structure and the production plan, the software will run and provide a detailed grid status at every time window. GridPenguin is proposed as a benchmark to evaluate and compare different optimization methods. The software has a more detailed model of a DHS than most models built for optimization and using it to evaluate the optimization methods provides better insights into the performance of the optimization. It also makes it possible to compare different methods. Second, as it gives users the freedom to build any grid they want, they can use it to generate data for machine learning methods. It is more accessible and flexible than a dataset collected from a real grid and the accuracy is good, compared with existing software on modelling heat/hydraulic grids.

There are a few existing software packages on the market and in the research community that can also model district heating networks. However, they are built with a different focus and have a different functionality and do not meet these goals. Wanda is a powerful software package which models a hydraulic system with good accuracy. Wanda is very accurate in the simulation of hydraulics but less so in heat transfer and heat production. This means it is not ideal for studies of heat networks where simulation of heat is generally more important than hydraulics. Wanda also runs relatively slow. ESSIM/ CHESS is an energy system simulator. At the moment of writing, the source code is not available and we could not find more detailed information about CHESS. DisHeatLab models the pipes and flows in a district heating system, but we could not find detailed models for heat exchangers or various types of producers. This limits the power of an optimizer that works using data from such a simulator. Comsof models the heat grid on high level and in a steady state. It cannot facilitate a detailed, hour-to-hour plan and it is not open-sourced, making it difficult for any self/machine-learning method to interact with it.

The rest of this paper is structured as the following: in Section 2, we give implementation details of the software, including the physical equations describing the behaviors, the computational methods to simulate these equations and the interface of modules of the software. In Section 3 we compare our model with the detailed hydraulic modelling software Wanda, and we show how accurate our model is compared to other similar software: Wanda. In Section 4, we discuss aspects that can be improved and what future work could focus on. Finally, the conclusion is given in Section 5.

2 MODEL IMPLEMENTATION

In this section we explain how the computational model is implemented: what physical relations are considered, how are they simulated computationally and in what order are they computed. With this section we hope to provide the reader a clear overview of what the model can be used for.

2.1 Main design choices

On the one hand, to serve as a benchmarking tool for district heating optimization algorithms, a simulator needs to have good accuracy. On the other hand, the need to use it intensively and repeatedly, as is required for example for machine learning algorithms [8], means it needs to have a decent speed as well. For the trade-off between speed and accuracy, we make the following design choices: 1. We aim to have a highly detailed simulation of heat and temperature. This includes the heat exchanging, heat loss and the production of heat; 2. Time is discretized almost everwhere in GridPenguin. The length of these time steps are specified by the user, for example at 15 minutes or 1 hour. The simulation accuracy should improve with smaller time steps, naturally at the cost of an increase in simulation time; 3. We simulate the pressure and the cost of running pumps in a simplified way for the speed of the simulation. Also, as reported in an earlier study, pressure is insignificant for optimizing the heat grid [7].

To represent the physical grid infrastructure in GridPenguin, we use two types of components: nodes and edges. An edge represents a pipe that is used for water transportation. A node may represent various components, such as a producer (CHP, heat pump, geothermal source, boiler), consumer, junction branch or water pump. Each of these components in GridPenguin is used in every step of the simulation.

In each time step, the computation starts from the consumers, then components that are directly connected to consumers, then components connected to those components. This continues through the whole grid and terminates when reaching the producers. Mass flow propagates through the grid with no time delay and total mass of water in the grid as well as water density is assumed constant.

To run GridPenguin, the following inputs are needed: the heat demand at each consumer (in MW), and the production plan of a producer in either heat production (in MW) or output temperature (in °C). If electricity trading is considered, then also the market electricity price and the electricity production plan (in MW) should be included. After the simulation, the user can obtain the output of almost every physical property that is relevant to a heat network, such as the temperature at any point of the grid, the heat loss in an edge, mass flow, whether production plan is feasible, etc. These values also come in discrete time for each time step. Note that for readability we have omitted the time step subscript in the equations in this section.

2.2 The Edge

The two physical equations involved in the simulation of water flowing through a pipe are the heat loss equation and the pressure loss equation. The simulation of heat loss to the environment uses the following equation [7]:

$$\mathrm{T_{end}} = (T_{start} - T_{env}) \cdot e^{-\frac{(\nabla t \cdot R_\lambda)}{A \cdot \rho \cdot c}} + T_{env} \quad (1)$$

Where T_{start}, T_{env} are the start temperature of the water and the environment in °C or K respectively, ∇t is the time difference between start and end, R_λ is the thermal resistance in W/m · K, A is the cross section area in m^2, ρ is water density and c is water heat capacity.

For pressure loss of mass flow due to friction along the pipe, we used a simplified model from Li et al. [3]: $\nabla p = f \cdot \dot{m}^2$ where ∇p is the pressure loss in pa, f is the friction coefficient in kg^{-1} · m^{-1} and $\dot{m}$ is the mass flow in kg/s.

An important cause of the complexity of modelling a district heat grid is the delay effect of the pipes. We model the dynamics in the edge/pipe using the node method [7]. An illustration of this method is provided in Figure 1.

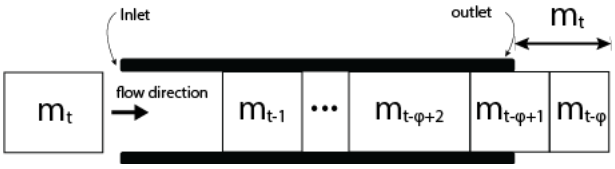

Figure 1: The simulation method of water in an edge. Water comes in at each time step is treated as a block with mass, entry temperature and entry step

In this method, water is treated as if it is a rigid block, called plug. A plug saves the information of mass, entry temperature and entry step. An edge is filled with plugs of which the mass always adds up to the mass that the edge can contain. A plug cannot be compressed and no heat exchange happens between plugs.

When solving the edge, the mass m_t has to be specified prior to the computation from either upstream or downstream. The plugs to be pushed out of/ into the edge are then calculated with the node method. The input, output and the constraint of an Edge are shown in Table 1.

Table 1 The input, output parameters and constraints of an Edge. *Delay matrix is to track, for example, for water flows out at t = 5, 40% are from t = 2 and 60% are from t = 3

Input parameters			
	Diameter	Length	R_λ (thermal resistance)
	c (heat capacity)	ρ (density)	T_{env}
	f (friction coef.)		
Output parameters			
	Inlet/outlet *T*	Mass flow	Delay matrix*
	Heat loss	Heat in pipe	Pressure loss
Constraints			
	Max flow speed	Min flow speed	

2.3 The Consumer

The relation between the heat the consumer received and the inlet, outlet temperature and mass flow is defined by:

$$\nabla Q = (T_{in} - T_{out}) \cdot \dot{m} \cdot c \tag{2}$$

A consumer always has a built-in heat exchanger (HX), as shown in Figure 2. Each consumer has a fixed inlet temperature at the consumer side of the heat exchanger (so not in the main grid) T'_{in}, and the consumer side outlet temperature T'_{out} is assumed to stay constant unless the demand is not met. Note that this is the model of a heat exchanger that has a control to guarantee a constant temperature at the consumer outlet side. The mass flow $\dot{m}'$is calculated so that the demand is met. Then we calculate T_{out} and $\dot{m}$ of the grid side using the NTU and LMTD methods [8]. When demand cannot be met, NTU is used. Otherwise, LMTD is used as it has better accuracy. Interested readers can refer to our source code for detailed implementation. Finally, the pressure load of the consumer is assumed to be a constant ∇p_c.

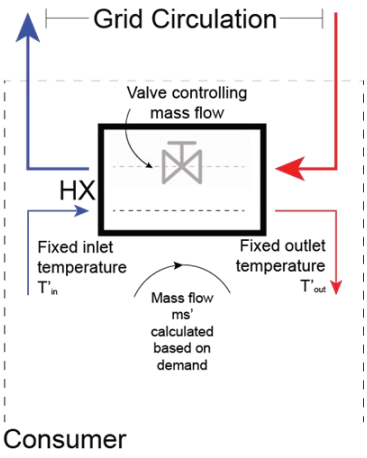

Figure 2: The structure of a consumer with built-in heat exchanger.

The computation of the consumer goes as follows: it retrieves an approximated inlet temperature T_{in} from the upstream component. Alternatively, it can retrieve a list of plugs: (temperature, mass) from the upstream pipe for better accuracy with more computation time and calculate the mass flow of each time during the time step. The current implementation uses the latter way. Then it calculates the average mass flow $\dot{m}$ in this time step and outlet temperature T_{out}. The input, output and constraints on a Consumer are shown in Table 2.

Table 2: The input, output parameters and constraints of a Consumer.

Input parameters			
	Demand	*c* (heat capacity)	$\dot{m}_{max}$
	A (surface area)	q, k(HX parameters)	∇p_c
	$T_{in,min}$	T'_{in}	T'_{out}
Output parameters			
	Inlet/outlet *T*	Mass flow	Delivered heat
Constraints			
	Demand satisfaction		

2.4 Producer (CHP)

A producer adds heat to the grid by consuming fuel or electricity and may produce electricity itself. The water pump is also modelled inside the producer. Currently we only have the Combined Heat and Power Plant (CHP) modelled, but other types can be added easily.

The key to model a CHP is an operating region and the corresponding cost. Most research in this field use a key-point approach or can be represented by a key-point approach [3] [4] [10] and this is what we implemented in GridPenguin. As shown in Figure 3, the user defines a convex operation region by a set of points $(H_1, P_1), (H_2, P_2)$. In addition, a cost array (α, β) must be defined so that the cost:

$$C = \alpha \cdot H + \beta \cdot P \tag{3}$$

where (H, P) is any point inside the operating region [11]. The CHP also has ramping constraints on heat, power and outlet temperature.

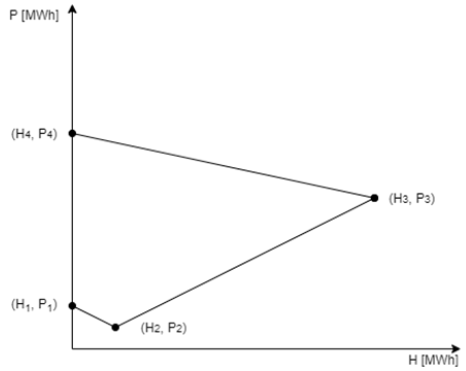

Figure 3: The operation region of the CHP, defined by 4 key points. H: heat production. P: power production.

The CHP can take either outlet temperature or heat production as input, together with power production. The resulting computations are slightly different. When the outlet temperature is the input, the downstream component can get an accurate outlet temperature from the CHP. However, when the input is the heat production, the downstream component does not know what temperature it is getting from the producer. So when it retrieves temperature from the producer, it also needs to provide information about an (approximated) mass flow.

The computation of the CHP itself is simple. If the outlet temperature is given, it calculates the heat production, or the other way around if the heat production is given. Next, it checks if any constraints are violated. Finally, it calculates a margin:

$$F = -\frac{C}{\eta} + \gamma \cdot P \tag{4}$$

where γ is the market electricity price and η is CHP efficiency. The input, output and constraints of a CHP are shown in Table 3.

Table 3: The input, output parameters and constraints of a CHP.

Input parameters			
	T_{out} *or* H	P (power production)	α, β (cost coef.)
	γ (electricity price)	∇p_c	pump efficiency
	ρ (density)	η (efficiency)	*c* (heat capacity)
Output parameters			
	Inlet/outlet *T*	Cost, profit and margin	Mass flow
	Heat, Power production		
Constraints			
	Heat, power, Temperature ramp	(H,E) inside Operation region	$T_{out} < T_{max}$

2.5 The Connector

The Connector is used to split one edge into two or more edges or to merge two or more edges into one. It is necessary for a non-trivial grid with more than one producer and/or more than one consumer.

In the case of one producer and multiple consumers, there will be no control variable for the Connector as the mass flow on the main edge is calculated by the sum of all splits. Take a connector for example with a producer upstream and 3 consumers downstream. The connector will be added 3 times to the solving list by 3 consumers. Each time when it is getting

solved, it checks if all mass flows of the downstream edges are known and only then it adds the upstream edge to the solving list.

In the case of multiple producers, the Connector that merges the producers together must decide how much mass is coming from each producer. As an example, take a connector which has two upstream producers and one downstream edge. The downstream edge will be solved first and the outlet mass flow of the branch will be known. A valve split parameter μ must be set by the user/ the optimizer. For example, if $\mu = (0.4,0.6)$ it means 40% of the mass comes from inlet 1 (producer 1) and 60% comes from inlet 2.

2.6 The Pump and Pressure Calculation

Because of the reasons we mentioned in section 2.1, we largely simplified the computation of pressure load and pressure propagation to have a better computation time and make it easier for users who do not have deep knowledge of hydraulic systems. We also do not care about pressure at any arbitrary point in the grid, but only the pressure load of the pump as only this is directly related to cost. The calculation makes sure that the pressure load, hence pump cost, is usually over-estimated. The over-estimation is preferred over under-estimation because then if it is feasible in the simulation, it should also be feasible in the reality.

We propagate the pressure loss, starting from the consumers, through the grid and end at the producers. The final pump cost is:

$$C_{pmp} = \frac{\nabla p \cdot \dot{m}}{\rho \cdot \eta_{pmp}} \tag{5}$$

where η_{pmp} is the efficiency of the pump and ∇p is the pressure difference at the producer/pump.

3 MODEL VALIDATION

To validate the simulator, we compared it with a more complex model for hydraulic systems: Wanda. It has been widely used in both academic and industrial areas for its accurate simulation. We compared two modules mainly: the edge and the heat exchanger (consumer). The producer module is not compared as Wanda does not include a detailed model of a producer/ CHP. The results show that GridPenguin has good accuracy on modelling the relation and propagation of heat, temperature and mass flow.

3.1 Heat loss

Wanda and GridPenguin measure heat loss in different ways. While GridPenguin uses constant heat capacity which introduces a small error, Wanda introduces an error with their discretization approach. It is difficult to say which approach is better. In this section, we set up a single pipe in both software packages. With designated inlet temperature and mass flow, the outlet temperature directly reflects the heat loss; this is therefore monitored.

The first thing we noticed is that while GridPenguin uses the thermal resistance R_λ as a parameter to model heat loss, Wanda uses the heat transfer coefficient U. We show that with $R_\lambda = \pi \cdot d \cdot U$ where d is pipe diameter, the heat loss calculated from them are nearly the same, with a difference less that 0.01%. To study the differences in changing mass flow and what it means to GridPenguin, some changing mass flow patterns are tested. These changing mass flows are generated using Perlin noise, with 3 different octaves (rate of changing). The

inlet temperature remains constant. The deviation of cumulative heat loss between Wanda and GridPenguin is almost negligible, as shown in Table 4.

Table 4: Heat loss difference at different mass flow changing rate

Flow speed changing rate ($*10^{-6} m/s^2$)	Difference (%)
4.95	0.0123
1.21	0.0693
0.46	0.340

3.2 Heat Exchanger

For the heat exchanging process, Wanda uses a single parameter: heat transfer coefficient U to represent the heat exchanger, while GridPenguin uses a more complex process to calculate U that is dependent on the mass flow. If we force the U to be the same, the differences between GridPenguin and Wanda outlet temperatures are smaller than 0.005 °C. Thus, we show that U is the only difference between Wanda and GridPenguin. We consider GridPenguin U to be more accurate as it takes mass flows into consideration.

In the real world, the heat exchanger is controlled with a set of logic modules and this comes with a delay. For example, if there is a sudden drop in the secondary outlet temperature, a PID will only open the valve on the primary side a little at a time, resulting in the secondary outlet temperature slowly recovering to the setpoint value. Wanda models the control like this. However, in GridPenguin, we think such small-scale changes are unnecessary, as in a real grid the control takes only few minutes or even seconds to adjust and this is thus insignificant to the time scale we model. So in GridPenguin the control will immediately react to the changing situation.

4 DISCUSSION AND FUTURE WORK

In this section we list the most relevant problems and improvements that future works can pay attention to: 1. The implementation of more complex storage modules and more producer types: geothermal, heat pump, boiler, etc. These are necessary to optimize grids with renewable energy sources, which of course become increasingly common; 2. A more accurate pressure load, especially at the valve of the HX. Although Benonysson [7] states that pump cost is insignificant, our experiments with the pressure do not fully agree with this statement. It might be interesting to implement a more accurate pressure model and try to see from there if pump cost is indeed insignificant; 3. The ability to handle a complex grid topology. The current implementation does not allow a bypass/ bi-directional pipe. However we know such structures exist in real-world grids; 4. The water physics can be more accurate. Currently we use constant density and heat capacity. However a more accurate model would make them temperature-dependent properties; 5. Studies on the speed and scalability of GridPenguin. How fast can the simulator run with different grid sizes? Does it scale more than linearly? 6. New control strategies of HX.

5 CONCLUSION

In the field of district heating optimization, there is a wide variety of methods being proposed, most of which involve mathematical optimization and model predictive control. However, these methods, when being proposed, are usually only compared with simple grid operation strategies as the result of the lack of a cross-board comparison criterion. To address

this problem, we propose GridPenguin. It is a powerful and comprehensive tool to simulate a district heating network in both good accuracy and high speed. We use discrete simulation with large time steps (typically 15-60 minutes) to ensure good speed while the node method and the modeling of a pipe with separate water plugs guarantee the accuracy. Details on small time scale are ignored while the energy balance and average flow/temperature on large time scale are calculated as accurately as possible. We use detailed physical equations for the calculation of heat loss and heat exchange. We also have included a detailed model of a producer. Compared with Wanda, a popular existing simulation tool, we show that GridPenguin has good accuracy and speed.

We present GridPenguin as a cross-board metric to evaluate and compare different optimization methods. Meanwhile, the computational speed allows the user to run a large number of simulation within limited time. This opens the possibility to apply data-driven machine learning methods, such as Reinforcement Learning [8], Tree Search and Deep Learning, in the DHS optimization problem.

6 ACKNOWLEDGEMENTS

We thank Ivo Pothof and Sam van der Zwan from Deltares for their discussion on the differences in simulation results between Wanda and GridPenguin.

7 REFERENCES

[1] L. Giraud, M. Merabet, R. Baviere and M. Vallee, “Optimal control of district heating systems using dynamic simulation and mixed integer linear programming,” in *Linkoping University Electronic Press*, 2017.

[2] R. Baviere and M. Vallee, “Optimal temperature control of large scale district heating networks,” *Energy Procedia,* vol. 149, pp. 69--78, 2018.

[3] Z. Li, W. Wu, M. Shahidehpour, J. Wang and B. Zhang, “Combined heat and power dispatch considering pipeline energy storage of district heating network,” *IEEE Transactions on Sustainable Energy,* vol. 7, pp. 12--22, 2015.

[4] L. Merkert, A. A. Haime and S. Hohmann, “Optimal scheduling of combined heat and power generation units using the thermal inertia of the connected district heating grid as energy storage,” *Energies,* vol. 12, p. 266, 2019.

[5] M. Casisi, S. Costanzo, P. Pinamonti and M. Reini, “Two-level evolutionary multi-objective optimization of a district heating system with distributed cogeneration,” *Energies,* vol. 12, p. 114, 2019.

[6] L. Urbanucci, F. D’Ettorre and D. Testi, “A comprehensive methodology for the integrated optimal sizing and operation of cogeneration systems with thermal energy storage,” *Energies,* vol. 12, p. 875, 2019.

[7] A. Benonysson, Dynamic modelling and operational optimization of district heating systems, 1991.

[8] K. Stepanovic, J. Wu, R. Everhardt and M. de Weerdt, “Unlocking the Flexibility of District Heating Pipeline Energy Storage with Reinforcement Learning,” *Energies,* vol. 15, p. 3290, 2022.

[9] T. Fang and R. Lahdelma, “Optimization of combined heat and power production with heat storage based on sliding time window method,” *Applied energy,* vol. 162, pp. 723--732, 2016.

[10] A. Bloess, “Modeling of combined heat and power generation in the context of increasing renewable energy penetration,” *Applied Energy,* vol. 267, p. 114727, 2020.

[11] H. V. Larsen, H. Palsson, B. Bohm and H. F. Ravn, “Equivalent models for district heating systems,” 1999.