Modelling Intelligent Energy Distribution Systems by Hyperdag P Systems

Adrian Zafiu[1], Cristian Ştefan[2]

[1] IMT Bucharest, Romania
[2] University of Piteşti, Romania
{adrian.zafiu,cristi.stefan}@upit.ro

Summary. The paper introduces a new model in membrane computing, using the hyperdag P systems to simulate a complex, feedback-driven energy distribution system. The proposed model is tested within an ad-hoc developed simulator, and the evolution of the system is presented step by step.

1 Introduction

The P systems are a computational model inspired from cellular biology, introduced by Păun [10] in 1998 in order to simulate the behaviour of natural systems by means of formal specifications.

Membrane computing is a vast research field, involving contributions from different areas, like parallel and distributed systems, financial case studies and evolution of living cells populations. There are many types of P systems, like tissue P systems, neural (spiking) P systems or asynchronous P systems. The model was further examined in Păun et al. [11].

The hyperdag P systems are a refinement of the original model, in which the tree structure is replaced by a directed acyclic graph (dag), introduced by Nicolescu [8] in 2008.

P system models allow realistic simulations of evolving systems, as transition rules can be applied separately for each cell, taking into account the environment factors (represented as promoters or inhibitors) and the received information from other cells (transported symbols).

2 Energy distribution systems (EDS) - a case study

When we consider an energy distribution system (EDS), there are two approaches. One involves the big-scale entities, like power plants, the national network of transformers and transmission lines, and finally the consumers (industrial-grade and home).

The small-scale approach regards the self-powered home, that has its own generators using renewable energy sources like the sun (photovoltaic panels) and the wind (eolian turbines).

The main goal in designing an ecological household regards the control of energy consumption level and the ways to optimize it.

In order to bypass short-time fluctuations that this kind of generators can suffer due to sudden changes in the environment factors, the system makes use of a set of batteries, that store the energy when it is available and give it back instantly if ecological power falls for a short period.

To summarize, the EDS (figure 1) has the following components:

- connection to the grid
- grid controller
- batteries
- battery regulator
- generators
- generator regulator
- the consumers
- the consumer controller (inverter)
- sensors
- sensor monitor
- memory for comfort variables
- main control unit (MCU)

As the natural factors change all the time, one cannot rely solely on independent generators to supply all the necessary power to a household for everyday needs. Thus, the connection to the national power grid is mandatory. To control how much power is taken from the grid and to monitor the costs involved, a grid controller is taken into account.

The set of batteries acts both as a buffer in case of short outages (temporary lack of wind or sunlight), and as an affordable alternative source for low-power requirements (night lighting, standby current for different devices), where grid energy can be avoided. Batteries charge only when there is enough *green power*, in order to keep the costs as low as possible. Their cycle is regulated by a dedicated controller to prevent overcharging or over-discharging, both being equally dangerous for the internal chemistry.

The generator set is the core of a self-powered home system, transforming the freely available energy from the natural sources, like sun and wind, into usable electrical power to drive all the devices that surround us and make our life easier. Using such energy implies reduced costs, long-term sustainability and a reduced impact on the planet's resources. The controller to which they are attached to is used for monitoring their usage, reporting failures and disconnecting them when power requirements indicate there's no need for more, in order to protect the life of moving components (turbine).

The consumers are all the electrical appliances that the owner makes use of but, for the case study in this paper, only the lighting and air conditioning systems are considered to be monitored and adjusted according to the desired parameters. Their controller has a function in conversion also, as the supplied DC voltage (usually 12 to 48V) from batteries and generators must be raised and converted to AC before it can be used.

The sensors read the instantaneous values (available light level and temperature) from the environment, and report them through their monitor to the MCU. They play the key role in the feed-back mechanism.

The desired values for the comfort variables (temperature and amount of light) that the user sets are stored in a dedicated memory that the MCU will read each time it needs to make an adjustment.

The MCU is the brain of the system, containing all necessary logic (rules) to request data from the memory and sensor controller, calculate the difference between values and issue the appropriate commands for the generators and consumers to adjust their behaviour as required. It is connected directly with all other controllers and the memory, as all communication between them passes through it.

3 Intelligent Energy distribution systems

3.1 The model structure and logic

The *intelligence* involved in the distribution is achieved through the rules implemented by the MCU, with the declared goal of minimizing the consumption from the grid. When the parameters need to be adjusted upwards, the first source considered are always the generators, as their energy comes almost for free (after the investment has been recovered). If they cannot supply the necessary instantaneous power, the second choice are the batteries, as they have an amount of power that comes also at no cost. If they are empty or have already reached the maximum that they can offer, there is no other option than to take the rest from the grid. This is the costly solution, but sometimes it's the only one left. When renewable power is available again, the first to be satisfied are the consumers, followed by the batteries who need to be refilled.

The philosophy behind such a system is to react promptly to the changes that occur and to satisfy the current needs without wasting energy when there's no one home, or at night, when there's usually no need for powerful lighting. If the temperature in the house is already at the desired level, the air conditioning system will not be started and, if the desired level has been reached after an increased consumption, the controller will just keep with it, without other increases.

3.2 Architecture - Hyperdag P systems

As mentioned in the Introduction, hyperdag P systems are a new family of P systems that R. Nicolescu proposed as an alternative to other existent types (tissue and neural P systems) in order to offer a more flexible way to communicate between cells, but respecting the hierarchical structure. In this approach the messages can be passed also to the cells on the same level (siblings), rewriting rules can be applied in a deterministic or parallel way, and the transfer modes can be dedicated (a single receiver) or spread across a domain (broadcast). The efficiency of such P systems has been proven by modelling problems like Synchronization in P Modules [2], the Byzantine Agreement [1] and optimizations to FSSP [4].

The basic definitions and notions from graph theory will not be discussed again here, as they can be found very easily in the literature.

The definition of hyperdag P systems and the two extensions are the ones given in Part A of the technical report by R. Nicolescu [7], [8], [9].

Definition 1. *A* hP system *(of degree m) is a system* $\Pi = (O, \sigma_1, \dots, \sigma_m, \delta, I_{out})$*, where:*

1. O *is an ordered finite non-empty alphabet of* objects*;*
2. $\sigma_1, \dots, \sigma_n$ *are* cells*, of the form* $\sigma_i = (Q_i, s_{i0}, w_{i0}, P_i), 1 \leq i \leq m$*, where:*
   - Q_i *is a finite set (of* states*),*
   - $s_{i0} \in Q_i$ *is the* initial state*,*
   - $w_{i0} \in O^*$ *is the* initial multiset *of objects,*
   - P_i *is a finite set of multiset rewriting* rules *of the form* $sx \to s'x'u_\uparrow v_\downarrow w_\leftrightarrow y_{go} z_{out}$*, where* $s, s' \in Q_i, x, x' \in O^*, u_\uparrow \in O^*_\uparrow, v_\downarrow \in O^*_\downarrow, w_\leftrightarrow \in O^*_\leftrightarrow, y_{go} \in O^*_{go}$*, and* $z_{out} \in O^*_{out}$ *with the restriction that* $z_{out} = \lambda$ *for all* $i \in \{1, \dots, m\} \setminus I_{out}$*;*
3. δ *is a set of dag parent/child arcs on* $\{1, \dots, m\}$*, i.e.,* $\delta \subseteq \{1, \dots, m\}x\{1, \dots, m\}$*, representing* bidirectional *communication channels between the cells;*
4. $I_{out} \subseteq \{1, \dots, m\}$ *indicates the* output cells*, the only cells allowed to send objects to the "environment".*

In addition to this definition, there are two more elements that should be presented in order to fully describe the simulation mechanism. One is the object transfer mode and the other is rewriting mode for symbols. Both define how rules are applied.

Regarding the object transfer mode, there are three options:

- *replication*: the replicated symbols are transmitted to all parents (↑), all children (↓) or all siblings (↔);
- *one*: the object will be delivered to a single, randomly chosen, parent (↑), child (↓) or sibling (↔);
- *spread*: the multiset will be decomposed and the parts are to be sent arbitrarily to the parents (↑), children (↓) or siblings (↔).

Regarding the symbol rewriting mode, there are also three options:

- *min*: the rule is applied once, if possible;
- *par*: rule is applied in parallel manner for all available symbols;
- *max*: a rule is applied as many times as possible.

It is important to mention that rules are applied in *weak priority* order, meaning that the ones with higher priority (appear at the beginning) come first, and that lower priority rules are applied only if they **do not** change the target state reached from the previous rules.

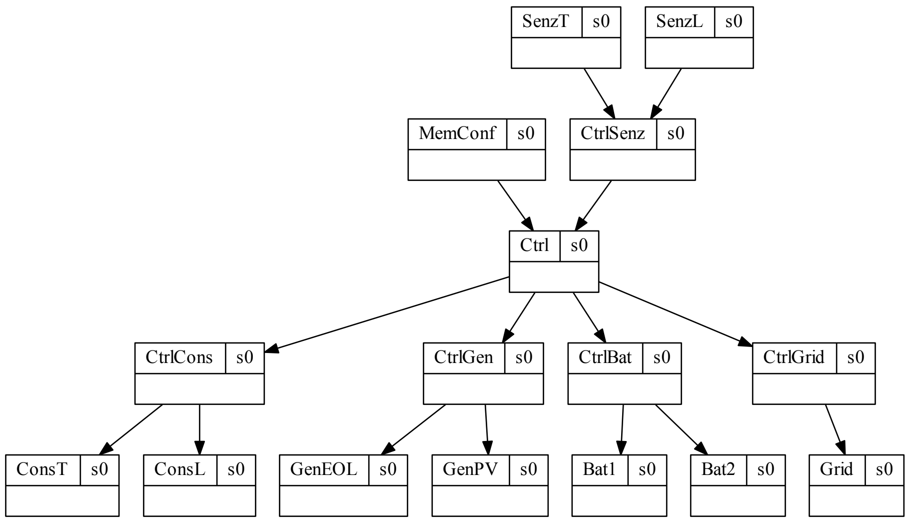

Fig. 1. General view of the system

3.3 Algorithm - The rule set and types

At the beginning, we define the connexions between the cells, by mentioning the parent and all its children. The initial cell configurations follow *@ lines*, as we define the memory for comfort variables *MemConf*, current u and maximum m energy values for each entity. Each consumer has a maximum amount of power that it can take, each generator has a limit of what it can give. The sensors store the values read from the environment. The first step is to send the trigger command q from the main control unit *Ctrl* to the memory and the sensors, in order to ask them to reply with their content. Sensors report to their dedicated controller *CtrlSenz*, which then sends the information to the main *Ctrl*.

After receiving all data, the controller is able to make the decision to increase or decrease the amount of energy offered by the generators, by calculating the difference between the desired temperature (or light) level - stored in *MemConf*, and the current one, reported by the sensors. The confirmation of energy availability will be sent to the consumers (temperature, lights), and they will increase their current consumption by one unit at each step.

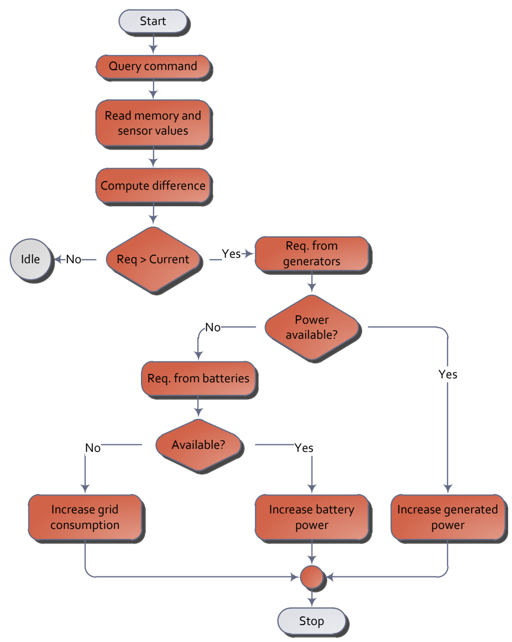

Fig. 2. Logical scheme

4 The simulator overview

The simulator implements the hyperdag P systems in respect with **Def. 1**, object rewriting rules and object transfer modes. At its core we defined the digraph structure, with arcs, nodes, rules, states and symbols as components, grouped into Configurations. We also implemented the *Rewriting* and *Transition types* as described above.

The direction for symbol transfer is indicated within the Behaviour class, all communication channels being considered as bi-directional. There are four options:

- *down*: symbols are sent to direct children of the current node;
- *up*: symbols are sent to the directly connected parent(s);
- *sibling*: objects reach the nodes on the same level and which are connected with the emitting cell;

- *out*: this is the production of the P system calculation, symbols are sent from the Output cell to the environment, and their multiplicity is regarded as the final result.

Rules form a separate class, each one having defined the initial and final states, the priority, rewriting and transition types. Rules are defined as strings entered by the user, as one would usually describe them, in the following form:

* **cell** *init.state* <sym._mult.> -> *f.state* <sym._mult._dir.rewr.transf.>

As an example, we present a rule for the *MCU*, which will propagate *down* the commands to increase the light and the power from the Grid, without changing the current state *sqc*. The rule is applied as many times as possible, and symbols are replicated to all the children:

$* \, Ctrl \; sqc \; ALu \; Su \rightarrow sqc \; ALu_{\downarrow} \; Su_{\downarrow} \; max \; repl$

Before entering the rules (marked by *), one needs to define the cells in the system, their connections and their initial states (lines beginning with @). When the command *Create* is given, the parser reads each line, builds the dag structure and the graphical representation on the fly (using *GraphViz* [3]) and loads each cell with its rule set. From that point, the system can evolve fully in one step (*Run* command) or step by step, with the currently applied rules being showed in red, for easier understanding of the transitions, and the content of each cell being updated in real time. Execution uses the parallel features of the .NET platform, cells that can evolve simultaneously have dedicated threads for their computations.

The simulator will be available for download in the near future at the following address: `http://fmi.upit.ro/psim/`.

5 Description of the rule set

In this section we present all the rules, grouped in subsections by each cell, and explain their roles in the system.

5.1 The memory for the comfort variables

For the cell *MemConf*, there are the following rules:

1. $s_0 \, q \; \rightarrow \; sqt \; min \; rep$
2. $sqt \; t \; \rightarrow \; sql \; t \; MT_{\downarrow} \; max \; rep$
3. $sqt \; \rightarrow \; sql \; min \; rep$
4. $sql \; l \; \rightarrow \; sqa \; l \; ML_{\downarrow} \; max \; rep$
5. $sql \; \rightarrow \; sqa \; min \; rep$
6. $sqa \; \rightarrow \; s_0 \; a_{\downarrow} \; max \; rep$

The meaning of the symbols are detailed in Table 1.

5.2 Main Control Unit

The *Ctrl* cell analyses and regulates the functioning of the entire system, and thus it has an increased number of rules and symbols. It communicates with all other Controllers in a full cycle. The states for this cell are, as follows:

Symbol	Description
q	The query request
t	The desired temperature value, stored in memory
l	The desired light level, stored in memory
MT	The response symbol for temperature
ML	The response symbol for light

Table 1. MemConf cell symbols

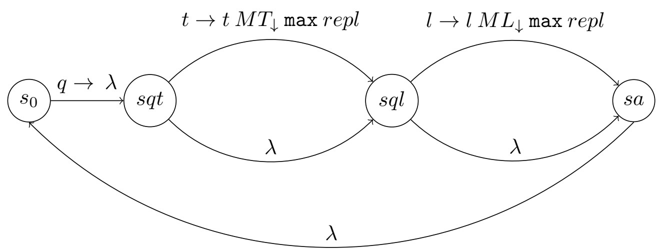

Fig. 3. State diagram for MemConf

s_0: initial state;
sq: waiting for a system query;
sqa: the analysis phase;
sqc: computing the base regulation (phase I), computing the commands for the full regulation (phase II) and cleaning the unnecessary symbols.

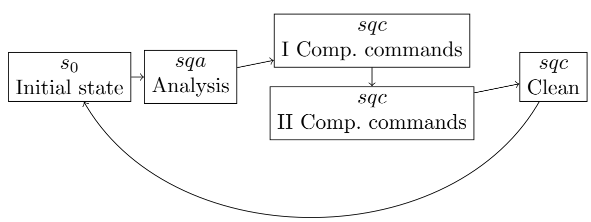

Fig. 4. State diagram for Ctrl

The role of the symbols are described in Table 2.

Symbol	Description
a	An answer
MT	The temperature value stored in memory
ST	The system temperature
CTu	Temperature can be increased
CTd	Temperature can be decreased
ATu	The computing answer is to increase the temperature
ATd	The computing answer is to decrease the temperature
ML	The light level stored in memory
SL	the light measured by the sensor
CLu	Light can be in increased
CLd	Light can be decreased
ALu	The computing answer is to increase the light
ALd	The computing answer is to decrease the light
GEu	The eolian generator has available power
GEd	The eolian generator cannot increase power
GPu	The photovoltaic generator has available power
GPd	The photovoltaic generator cannot increase power
B1u	Battery 1 can be charged
B1d	Battery 1 can be used for power
B2u	The battery 2 can be charged
B2d	The battery 2 can be used for power
Su	The Grid can offer more power
Sd	The Grid cannot offer more power

Table 2. Ctrl cell symbols

Request rules:

1. $s_0 \rightarrow sq\, q_{\updownarrow}\, min\, rep$
2. $sq\, a_6 \rightarrow sqa\, min\, rep$

The first step when starting the system is to send a query command (q) to all components. After receiving six answers (a symbols), the *Ctrl* has all necessary informations and it can start computing the differences and adjust the parameters by sending the appropriate commands to the other controllers.

The state analysis:

1. $sqa\, MT\, ST \rightarrow sqa\, max\, rep$
2. $sqa\, MT\, CTu \rightarrow sqa\, ATu\, min\, rep$
3. $sqa\, ST\, CTd \rightarrow sqa\, ATd\, min\, rep$
4. $sqa\, MT \rightarrow sqa\, max\, rep$
5. $sqa\, ST \rightarrow sqa\, max\, rep$
6. $sqa\, CTu \rightarrow sqa\, min\, rep$
7. $sqa\, CTd \rightarrow sqa\, min\, rep$
8. $sqa\, ML\, SL \rightarrow sqa\, max\, rep$
9. $sqa\, ML\, CLu \rightarrow sqa\, ALu\, min\, rep$
10. $sqa\, SL\, CLd \rightarrow sqa\, ALd\, min\, rep$
11. $sqa\, ML \rightarrow sqa\, max\, rep$
12. $sqa\, SL \rightarrow sqa\, max\, rep$
13. $sqa\, CLu \rightarrow sqa\, min\, rep$
14. $sqa\, CLd \rightarrow sqa\, min\, rep$

Rule #1 computes the difference between the desired Temperature value that is stored in Memory (MT) and the current one, read by the Sensor (ST). If there are MT symbols left and it is possible to increase the consumption for the heating device (*CTu* is present), the command is issued by creating the *ATu* symbol. If there are ST symbols present and it is possible to reduce the consumption (CTd is present), then we produce the command to decrease the Temperature, *ATd*. Rules 4 to 7 clean the remaining symbols.

Rule #8 makes the difference between the desired Light level form the Memory (*ML*) and the current one from the Sensor (*SL*). If there are *ML* symbols present and it is possible to increase the power for lighting (*CLu* is present), then we produce the command ALu with rule #9. If there are *SL* symbols left, and it is possible to reduce the power for lighting (presence of *CLd*), then symbol *Ald* is produced. Rules #11. . . #14 do the cleaning.

The first set of regulation rules is the following:

1. $sqa \rightarrow sqc\, min\, rep$
2. $sqc\, B1o\, Sd \rightarrow sqc\, B1o_{\downarrow}\, Sd_{\downarrow}\, max\, rep$
3. $sqc\, B2o\, Sd \rightarrow sqc\, B2o_{\downarrow}\, Sd_{\downarrow}\, max\, rep$
4. $sqc\, GEu\, Sd \rightarrow sqc\, GEu_{\downarrow}\, Sd_{\downarrow}\, max\, rep$
5. $sqc\, GPu\, Sd \rightarrow sqc\, GPu_{\downarrow}\, Sd_{\downarrow}\, max\, rep$
6. $sqc\, GEu\, B1i \rightarrow sqc\, GEu_{\downarrow}\, B1i_{\downarrow}\, max\, rep$
7. $sqc\, GPu\, B1i \rightarrow sqc\, GPu_{\downarrow}\, B1i_{\downarrow}\, max\, rep$
8. $sqc\, GEu\, B2i \rightarrow sqc\, GEu_{\downarrow}\, B2i_{\downarrow}\, max\, rep$
9. $sqc\, GPu\, B2i \rightarrow sqc\, GPu_{\downarrow}\, B2i_{\downarrow}\, max\, rep$

Rule #1 puts the *Ctrl* cell in a state where potential anomalies are detected and removed (rules #2. . . #9).

2, 3: If the Batteries can give more power (presence of *B1o* or *B2o*) and the consumption from the Grid can be decreased (presence of *Sd*), then the appropriate commands will be propagated down.

4, 5: If the Generators can give more power (*GEu* or *GPu* are present) and the consumption from the Grid can be decreased (*Sd* is there), then the commands are sent down.

6, 7, 8, 9: If the battery charge current can be increased (we have *B1i* or *B2i*) and the Generators can offer more energy, the appropriate commands are sent to their controller.

The second set of regulation rules consists of:

1. $sqc\, ATu\, GEu \rightarrow sqc\, ATu_{\downarrow}\, GEu_{\downarrow}\, max\, rep$
2. $sqc\, ATu\, GPu \rightarrow sqc\, ATu_{\downarrow}\, GPu_{\downarrow}\, max\, rep$
3. $sqc\, ATu\, B1o \rightarrow sqc\, ATu_{\downarrow}\, B1o_{\downarrow}\, max\, rep$
4. $sqc\, ATu\, B2o \rightarrow sqc\, ATu_{\downarrow}\, B2o_{\downarrow}\, max\, rep$
5. $sqc\, ATu\, Su \rightarrow sqc\, ATu_{\downarrow}\, Su_{\downarrow}\, max\, rep$
6. $sqc\, ATd\, Sd \rightarrow sqc\, ATd_{\downarrow}\, Sd_{\downarrow}\, max\, rep$
7. $sqc\, ATd\, B1i \rightarrow sqc\, ATd_{\downarrow}\, B1i_{\downarrow}\, max\, rep$
8. $sqc\, ATd\, B2i \rightarrow sqc\, ATd_{\downarrow}\, B2i_{\downarrow}\, max\, rep$
9. $sqc\, ATd\, GEg \rightarrow sqc\, ATd_{\downarrow}\, GEd_{\downarrow}\, max\, rep$

10. $sqc\, ATd\, GPg \rightarrow sqc\, ATd_{\downarrow}\, GPd_{\downarrow}\, max\, rep$
11. $sqc\, ALu\, GEu \rightarrow sqc\, ALu_{\downarrow}\, GEu_{\downarrow}\, max\, rep$
12. $sqc\, ALu\, GPu \rightarrow sqc\, ALu_{\downarrow}\, GPu_{\downarrow}\, max\, rep$
13. $sqc\, ALu\, B1o \rightarrow sqc\, ALu_{\downarrow}\, B1o_{\downarrow}\, max\, rep$
14. $sqc\, ALu\, B2o \rightarrow sqc\, ALu_{\downarrow}\, B2o_{\downarrow}\, max\, rep$
15. $sqc\, ALu\, Su \rightarrow sqc\, ALu_{\downarrow}\, Su_{\downarrow}\, max\, rep$
16. $sqc\, ALd\, Sd \rightarrow sqc\, ALd_{\downarrow}\, Sd_{\downarrow}\, max\, rep$
17. $sqc\, ALd\, B1i \rightarrow sqc\, ALd_{\downarrow}\, B1i_{\downarrow}\, max\, rep$
18. $sqc\, ALd\, B2i \rightarrow sqc\, ALd_{\downarrow}\, B2i_{\downarrow}\, max\, rep$
19. $sqc\, ALd\, GEg \rightarrow sqc\, ALd_{\downarrow}\, GEd_{\downarrow}\, max\, rep$
20. $sqc\, ALd\, GPg \rightarrow sqc\, ALd_{\downarrow}\, GPd_{\downarrow}\, max\, rep$

These rules are in charge of the increase (*ATu*, *ALu*) and decrease commands (*ATd*, *ALd*) for the Temperature and Light levels.

1..5: We try to increase the consumption for the Temperature (*ATu*) by checking the available sources, in the following order: Eolian Generator (*GEu*), Photovoltaic Generator (*GPu*), and the Batteries (*B1o*, *B2o*) and finally, as a last resort, the national power Source (*Su*). If any one of those has available power, the appropriate commands are sent to it.

6..10: We try to decrease the consumption for the Temperature and take into account the sources in reverse order: grid (*Sd*), batteries (*B1d*, *B2d*) and the generators (*GEd*, *GPd*). If the amount taken form any of these sources can be decreased, the commands are to be sent accordingly.

10..15: We try to increase the consumption for the Light (*ALu*) by checking the available sources, in the following order: Eolian Generator (*GEu*), Photovoltaic Generator (*GPu*), and the Batteries (*B1o*, *B2o*) and finally, as a last resort, the national power Source (*Su*). If any one of those has available power, the appropriate commands are sent to it.

16..20: We try to decrease the consumption for the Light and take into account the sources in reverse order: grid (*Sd*), batteries (*B1d*, *B2d*) and the generators (*GEd*, *GPd*). If the amount taken form any of these sources can be decreased, the commands are to be sent accordingly.

We use the following rules for cleaning:

1. $sqc\, MT \rightarrow sqc\, max\, rep$
2. $sqc\, ST \rightarrow sqc\, max\, rep$
3. $sqc\, ML \rightarrow sqc\, max\, rep$
4. $sqc\, SL \rightarrow sqc\, max\, rep$
5. $sqc\, B1i \rightarrow sqc\, max\, rep$
6. $sqc\, B1o \rightarrow sqc\, max\, rep$
7. $sqc\, Su \rightarrow sqc\, max\, rep$
8. $sqc\, Sd \rightarrow sqc\, max\, rep$
9. $sqc\, GEu \rightarrow sqc\, max\, rep$
10. $sqc\, GEd \rightarrow sqc\, max\, rep$
11. $sqc\, GPu \rightarrow sqc\, max\, rep$
12. $sqc\, GPd \rightarrow sqc\, max\, rep$
13. $sqc\, B2i \rightarrow sqc\, max\, rep$
14. $sqc\, B2o \rightarrow sqc\, max\, rep$
15. $sqc\, CTu \rightarrow sqc\, max\, rep$
16. $sqc\, CTd \rightarrow sqc\, max\, rep$
17. $sqc\, CLu \rightarrow sqc\, max\, rep$
18. $sqc\, CLd \rightarrow sqc\, max\, rep$
19. $sqc \rightarrow s_0$

In the end, all unused symbols from this cell are cleared, rule #19 having the role to prepare the cell for a new computation cycle.

5.3 The sensor controller

For the cell *CtrlSens* we defined the following rules:

1. $s_0\, q \;\to\; s_0\, q_{\uparrow}\, min\, rep$
2. $s_0\, a_2 \;\to\; sq\, a\, min\, rep$
3. $sq\, ST \;\to\; sq\, ST_{\downarrow}\, max\, rep$
4. $sq\, SL \;\to\; sq\, SL_{\downarrow}\, max\, rep$
5. $sq\, a \;\to\; s_0\, a_{\downarrow}\, min\, rep$

Description for these rules is given below:

1: the response request is forwarded to the sensors;
2: the cell waits for all sensors to answer;
3, 4: the measured values are relayed to the general controller;
5: cell confirms that all measured values have been submitted.

If the Temperature sensor receives a query, it will answer with an ST symbol for each t, and the Light sensor will answer with an SL symbol for each l it contains.

The description of the symbols is given in the table 3.

Symbol	Description
q	Query
t	The measured temperature
ST	The response symbol for temperature
l	The measured light
SL	The response symbol for light

Table 3. Sensor Controller symbols

5.4 Sensors

As announced, we have two sensors that measure the current temperature and amount of light from the environment we wish to monitor and control.

The rules for the Temperature sensor ($SensT$) are the following:

1. $s_0\, q \;\to\; sqt\, min\, rep$
2. $sqt\, t \;\to\; sqa\, t\, ST_{\downarrow}\, max\, rep$
3. $sqt \;\to\; sqa\, min\, rep$
4. $sqa \;\to\; s_0\, a_{\downarrow}\, min\, rep$

For the Light sensor *SensL* we have:

1. $s_0\, q \;\rightarrow\; sqt\; min\, rep$
2. $sql\, t \;\rightarrow\; sqa\, l\, SL_{\downarrow}\; max\, rep$
3. $sql \;\rightarrow\; sqa\; min\, rep$
4. $sqa \;\rightarrow\; s_0\, a_{\downarrow}\; min\, rep$

Again, they will answer with ST and SL for the queries, the same as their Controller, symbols having the same sense.

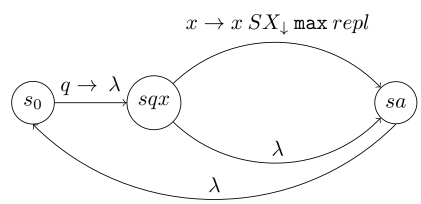

Fig. 5. State diagram for the Sensors

5.5 Consumers controller

The main property for the consumers is the availability to increase or decrease their current power absorbed. Let variable X represent the consumers. The request received by the consumer controller will be sent to all consumers. They will answer with *CXu*, meaning that the consumption for variable X (where $X \in \{Temp., Light\}$) can be increased, or with *CXd*, meaning that the consumption for that variable can be decreased. These symbols are further forwarded to the MCU, when answers from all consumers have been received.

The second phase regards treating the commands for actually increasing or decreasing the consumptions. These are forwarded to the consumers themselves for execution. Unnecessary symbols are then cleared.

Rules for the cell *CtrlCons* are the following:

1. $s_0\, q \;\rightarrow\; sq\, q_{\downarrow}\; min\, rep$
2. $sq\, a_2 \;\rightarrow\; sa\; min\, rep$
3. $sa\, CTu \;\rightarrow\; sa\, CTu_{\uparrow}\; min\, rep$
4. $sa\, CTd \;\rightarrow\; sa\, CTd_{\uparrow}\; min\, rep$
5. $sa\, CLu \;\rightarrow\; sa\, CLu_{\uparrow}\; min\, rep$
6. $sa\, CLd \;\rightarrow\; sa\, CLd_{\uparrow}\; min\, rep$
7. $sa \;\rightarrow\; s_0\, a_{\uparrow}\; min\, rep$
8. $s_0\, ATu \;\rightarrow\; s_0\, ATu_{\downarrow}\; max\, rep$
9. $s_0\, ATd \;\rightarrow\; s_0\, ATd_{\downarrow}\; max\, rep$
10. $s_0\, ALu \;\rightarrow\; s_0\, ALu_{\downarrow}\; max\, rep$
11. $s_0\, ALd \;\rightarrow\; s_0\, ALd_{\downarrow}\; max\, rep$
12. $s_0\, GEu \;\rightarrow\; s_0\; max\, rep$
13. $s_0\, GEd \;\rightarrow\; s_0\; max\, rep$
14. $s_0\, GPu \;\rightarrow\; s_0\; max\, rep$
15. $s_0\, GPd \;\rightarrow\; s_0\; max\, rep$
16. $s_0\, B1o \;\rightarrow\; s_0\; max\, rep$
17. $s_0\, B1i \;\rightarrow\; s_0\; max\, rep$
18. $s_0\, B2o \;\rightarrow\; s_0\; max\, rep$
19. $s_0\, B2i \;\rightarrow\; s_0\; max\, rep$
20. $s_0\, Su \;\rightarrow\; s_0\; max\, rep$
21. $s_0\, Sd \;\rightarrow\; s_0\; max\, rep$

Description for these rules is given below:

1: the request is transmitted to all consumers;
2: the cell waits for all consumers to answer;
3..6: the answers are relayed to the general controller;
7: the cell confirms that all answers have been submitted;
8..11: all commands are distributed to the consumers;
12..21: clean the unnecessary symbols.

The symbols' meanings are shown in Table 4.

Symbol	Description
q	Query
a	Counting the consumers answers
CTu	Temperature can be increased
CTd	Temperature can be decreased
ATu	The computing answer is to increase the temperature
ATd	The computing answer is to decrease the temperature
CLu	Light can be in increased
CLd	Light can be in decreased
ALu	The computing answer is to increase the light
ALd	The computing answer is to decrease the light

Table 4. CtrlCons cell symbols

5.6 Consumers

Each consumer can be asked to report it's actual state. The state consists of it's current consumption level (the multiplicity of the symbol u, if $u > 0$) and the availability to increase it (if $u < m$, m being the maximum value). The answer can be positive or negative. The second set of commands is about actually increasing the consumption, which is executed. Unknown commands are to be cleared.

The description is given for a generic consumer, indicated by X. The rules for a consumer cell are the following:

1. $s_0\, q \;\rightarrow\; sqd\, min\, rep$
2. $sqd\, u \;\rightarrow\; squ\, u\, ud\, min\, rep$
3. $sqd \;\rightarrow\; squ\, min\, rep$
4. $squ\, u\, m \;\rightarrow\; squ\, d\, max\, rep$
5. $squ\, m \;\rightarrow\; sqa\, m\, uu\, min\, rep$
6. $sqa\, d \;\rightarrow\; sqa\, u\, m\, max\, rep$
7. $sqa\, uu \;\rightarrow\; sqa\, CXCu_{\uparrow}\, min\, rep$
8. $sqa\, ud \;\rightarrow\; sqa\, CXd_{\uparrow}\, min\, rep$
9. $sqa \;\rightarrow\; s0\, a_{\uparrow}\, min\, rep$
10. $s0\, AXu \;\rightarrow\; s0\, u\, max\, rep$
11. $s0\, AXd\, u \;\rightarrow\; s0\, max\, rep$

The rules description is given below:

1: request to for the consumer state;
2..6: computing the difference between maximum and current values;

7, 8: the cell emits the answer;
9: cell confirms that answers were submitted;
10: the cell increases consumption;
11: the cell decreases consumption.

The symbols' meaning are shown in Table 5.

Symbol	Description
u	The consumption
m	The maximum value for the consumption
q	Query
a	Answer acknowledge
CXu	Consumption can be increased
CXd	Consumption can be decreased
AXu	Request to increase the consumption
AXd	Request to decrease power

Table 5. Consumer cell symbols

5.7 Generators controller

The generators have or not the ability to increase or decrease the amount of power they give at each moment. Let X be the generic name for a generator. The request received by the generator controller is further spread to all generators defined. They will answer with either *GXu*, meaning that they can increase the power, or *GXd*, if they can decrease their power. These symbols are to be delivered to the MCU when all generators have sent their answers.

The second phase of using a generator occurs when commands for increasing or decreasing the given power are actually received. These are forwarded to the generators, and the unnecessary symbols are to be cleaned up from their Controller.

The specific rules for the *CtrlGen* cell are as follows:

1. $s_0\, q \;\rightarrow\; sq\, q_{\downarrow}\, min\, rep$
2. $sq\, a_2 \;\rightarrow\; sa\, min\, rep$
3. $sa\, GEu \;\rightarrow\; sa\, GEu_{\uparrow}\, max\, rep$
4. $sa\, GEd \;\rightarrow\; sa\, GEd_{\uparrow}\, max\, rep$
5. $sa\, GPu \;\rightarrow\; sa\, GPu_{\uparrow}\, max\, rep$
6. $sa\, GPd \;\rightarrow\; sa\, GPd_{\uparrow}\, max\, rep$
7. $sa \;\rightarrow\; s_0\, a_{\uparrow}\, max\, rep$
8. $s_0\, GEu \;\rightarrow\; s_0\, GEu_{\downarrow}\, max\, rep$
9. $s_0\, GEd \;\rightarrow\; s_0\, GEd_{\downarrow}\, max\, rep$
10. $s_0\, GPu \;\rightarrow\; s_0\, GPu_{\downarrow}\, max\, rep$
11. $s_0\, GPd \;\rightarrow\; s_0\, GPd_{\downarrow}\, max\, rep$
12. $s_0\, ATu \;\rightarrow\; s_0\, max\, rep$
13. $s_0\, ATd \;\rightarrow\; s_0\, max\, rep$
14. $s_0\, ALu \;\rightarrow\; s_0\, max\, rep$
15. $s_0\, ALd \;\rightarrow\; s_0\, max\, rep$
16. $s_0\, B1o \;\rightarrow\; s_0\, max\, rep$
17. $s_0\, B1i \;\rightarrow\; s_0\, max\, rep$
18. $s_0\, B2o \;\rightarrow\; s_0\, max\, rep$
19. $s_0\, B2i \;\rightarrow\; s_0\, max\, rep$
20. $s_0\, Su \;\rightarrow\; s_0\, max\, rep$
21. $s_0\, Sd \;\rightarrow\; s_0\, max\, rep$

Description for these rules is given below:

1: the response request is forwarded to the generators;
2: the cell waits for all generators to answer;
3..6: the measured values are relayed to the general controller;
7: cell confirms that all measured values were submitted;
8..11: all commands are submitted to generators;
12..21: cleaning rules.

The symbols *GXu* and *GXd* play two roles:

1. if the cell is in the state *sa*, then the symbol indicating a generator state (G) is sent to the general controller;
2. if the cell is in state s_0, then the symbol designating a command forwarded to the each generator.

Unlike the consumers, symbols *GXu* and *GXd* have the multiplicity equal with the number of units that the power amount can be increased or decreased with.

The symbols' meaning are shown in Table 6.

Symbol	Description
q	Query
a	Answer acknowledge
GEu	Eolian generator can increase power
GEd	Eolian generator decrease power
GPu	Photovoltaic generator can offer more
GPd	Photovoltaic generator can offer less

Table 6. CtrlGen cell symbols

5.8 Generators

Each generator can be queried about it's current state. The state is about the actual power it gives, indicated by the multiplicity of the symbol u, and the availability to increase that power if the maximum value (m) has not been reached. The answer can be positive or negative.

The second mode for the generators occurs when they receive actual power increase or decrease commands, which are to be executed directly. Unnecessary commands need to be cleared.

The description is given for a generic generator, indicated by X:

The rules for a generator are the following:

1. $s_0\, q \;\rightarrow\; sqd\, min\, rep$
2. $sqd\, u \;\rightarrow\; squ\, u\, ud\, max\, rep$
3. $sqd \;\rightarrow\; squ\, min\, rep$
4. $squ\, u\, m \;\rightarrow\; squ\, d\, max\, rep$
5. $squ\, m \;\rightarrow\; sqa\, m\, uu\, max\, rep$
6. $sqa\, d \;\rightarrow\; sqa\, u\, m\, max\, rep$
7. $sqa\, uu \;\rightarrow\; sqa\, GXu_{\uparrow}\, max\, rep$
8. $sqa\, ud \;\rightarrow\; sqa\, GXd_{\uparrow}\, max\, rep$
9. $sqa \;\rightarrow\; s_0\, a_{\uparrow}\, min\, rep$
10. $s_0\, GXu \;\rightarrow\; s_0\, u\, max\, rep$
11. $s_0\, GXd\, u \;\rightarrow\; s_0\, max\, rep$

Rules description:

1: the request to report the consumer state;
2..6: computing the difference between maximum and current values;
7, 8: cell returns the answer;
9: the cell confirms that answers were submitted;
10: the cell increases generated power;
11: the cell decreases energy offered.

The symbols' meaning are shown in Table 7.

Symbol	Description
u	The actual power level
m	The maximum power
q	Query
a	Answer acknowledge
GXu	Request to increase the generated power
GXd	Request to decrease the generated power

Table 7. Generator cell symbols

5.9 Battery controller

Batteries are defined by the availability to increase or decrease the power they offer at each instant. The request to the *CtrlBat* is further disseminated to all batteries. They will answer each with *BXi* - the value with which the charge current can be increased, or *BXo* - the value with which the amount of power they give can be increased, where $X in \{1 \ldots n\}$. These symbols are further relayed to the MCU when answers from all batteries have been received.

Rules for *CtrlBat* are as follows:

1. $s_0\, q \;\rightarrow\; sq\, q_{\downarrow}\, min\, rep$
2. $sq\, a_2 \;\rightarrow\; sa\, min\, rep$
3. $sa\, B1i \;\rightarrow\; sa\, B1i_{\uparrow}\, max\, rep$
4. $sa\, B1o \;\rightarrow\; sa\, B1o_{\uparrow}\, max\, rep$
5. $sa\, B2i \;\rightarrow\; sa\, B2i_{\uparrow}\, max\, rep$
6. $sa\, B2o \;\rightarrow\; sa\, B2o_{\uparrow}\, max\, rep$
7. $sa \;\rightarrow\; s_0\, a_{\uparrow}\, max\, rep$
8. $s_0\, B1o \;\rightarrow\; s_0\, B1o_{\downarrow}\, max\, rep$
9. $s_0\, B1i \;\rightarrow\; s_0\, B1i_{\downarrow}\, max\, rep$
10. $s_0\, B2o \;\rightarrow\; s_0\, B2o_{\downarrow}\, max\, rep$
11. $s_0\, B2i \;\rightarrow\; s_0\, B2i_{\downarrow}\, max\, rep$
12. $s_0\, ATu \;\rightarrow\; s_0\, max\, rep$

13. $s_0\,ATd \rightarrow s_0\,max\,rep$
14. $s_0\,ALu \rightarrow s_0\,max\,rep$
15. $s_0\,ALd \rightarrow s_0\,max\,rep$
16. $s_0\,GEu \rightarrow s_0\,max\,rep$
17. $s_0\,GEd \rightarrow s_0\,max\,rep$
18. $s_0\,GPu \rightarrow s_0\,max\,rep$
19. $s_0\,GPd \rightarrow s_0\,max\,rep$
20. $s_0\,Su \rightarrow s_0\,max\,rep$
21. $s_0\,Sd \rightarrow s_0\,max\,rep$

Rules description:

1: the request is retransmitted to the batteries;
2: cell waits for all batteries to answer;
3..6: measured values are relayed to the general controller;
7: cell confirms that all measured values were submitted;
8..11: all commands are submitted to batteries;
12..21: cleaning rules.

The symbols *BXi* and *BXo* play two roles:

1. if the cell is in the state *sa*, then the symbol indicating a battery state (B) is sent to the general controller;
2. if the cell is in state s_0, then the symbol designating a command forwarded to the each battery.

Unlike the consumers, symbols *BXi* and *BXo* have the multiplicity equal with the number of units that the power amount taken or given (for charging) can be increased or decreased with.

The symbols' meaning are shown in Table 8.

Symbol	Description
q	Query
a	Answer acknowledge
BXi	The battery X can give more energy
BXo	The battery X can charge more

Table 8. Battery controller symbols

5.10 Batteries

Each battery can be queried about its current state. The actual state refers to:

u, m: Energy level and maximum level;
iu, im: Charge level and maximum charge;
du, dm: Discharge level and maximum discharge.

A battery can have dual behaviour: can be a generator as it discharges, but becomes a consumer when it charges back. The two states differ by the maximum instantaneous values. Thus, the maximum amount with which the charging can be done is given by the stopping of discharging and maximizing the charge current. Also, the maximum available power is given when the charge current is 0.

The second state for the batteries occurs when they receive commands to increase or decrease their given power amount. The command translates in changes for the values *iu* and *du*. The unnecessary commands are to be removed.

The description is given for a generic battery, indicated by X.

Rules for cell Battery X are:

1. $s_0\, iu\, du \rightarrow s_0\, max\, rep$
2. $s_0\, q \rightarrow sqd\, min\, rep$
3. $sqd\, u\, m \rightarrow sqd\, d\, max\, rep$
4. $sqd\, iu\, im \rightarrow sqd\, id\, BXo_{\uparrow}\, max\, rep$
5. $sqd\, du\, dm \rightarrow sqd\, dd\, BXi_{\uparrow}\, max\, rep$
6. $sqd\, m \rightarrow sqi\, m\, min\, rep$
7. $sqd\, d \rightarrow sqo\, d\, min\, rep$
8. $sqd \rightarrow sqa\, min\, rep$
9. $sqi\, im \rightarrow sqo\, im\, BXi_{\uparrow}\, max\, rep$
10. $sqi \rightarrow sqo\, min\, rep$
11. $sqo\, dm \rightarrow sqa\, dm\, BXo_{\uparrow}\, max\, rep$
12. $sqo \rightarrow sqa\, min\, rep$
13. $sqa\, d \rightarrow sqa\, u\, m\, max\, rep$
14. $sqa\, id \rightarrow sqa\, iu\, im\, max\, rep$
15. $sqa\, dd \rightarrow sqa\, du\, dm\, max\, rep$
16. $sqa \rightarrow s_0\, a_{\uparrow}\, min\, rep$
17. $s_0\, BXo \rightarrow s_0\, du\, max\, rep$
18. $s_0\, BXi \rightarrow s_0\, iu\, max\, rep$

Rules description:

1: request to submit the battery state;
2..7: computing the difference between maximum value and inverse value for charge and discharge (decrease the inverse flow);
8, 12: computing the direct values for charge and discharge (increase the direct flow);
13..15: restore the initial cell's values;
16: cell confirms that answers were submitted;
10: the cell increases output flow;
11: the cell decreases input flow.

The symbols' meaning are shown in Table 9.

Symbol	Description
u	The available energy level
m	The maximum energy level
iu	The charge level
im	The maximum charge level
du	The discharge level
dm	The maximum discharge level
q	Query
a	Answer acknowledge
BXi	The input can be increased
BXd	The output can be increased

Table 9. Battery cell symbols

5.11 Grid controller

The power Grid is defined by the installed power (m) which can be taken from the physical line. Let X represent the source identifier (one can have many connections for different voltages, like 220V and 380V). The request from the Grid Controller is forwarded to the source itself. Each of them answers with *SXu*, if the amount of power can be increased, or *SXd*, if the power can be decreased. The symbols are then delivered to the MCU, when all answers from the sources have been received.

The second phase of using the Controller occurs when actual increase or decrease commands are received. These are forwarded to the sources themselves, and all unnecessary symbols are cleaned up.

Rules for the cell *CtrlGrid* are presented below:

1. $s_0\, q \;\rightarrow\; sq\, q_{\downarrow}\; min\, rep$
2. $sq\, a \;\rightarrow\; sa\; min\, rep$
3. $sa\, Su \;\rightarrow\; sa\, Su_{\uparrow}\; max\, rep$
4. $sa\, Sd \;\rightarrow\; sa\, Sd_{\uparrow}\; max\, rep$
5. $sa \;\rightarrow\; s_0\, a_{\uparrow}\; max\, rep$
6. $s_0\, Su \;\rightarrow\; s_0\, Su_{\downarrow}\; max\, rep$
7. $s_0\, Sd \;\rightarrow\; s_0\, Sd_{\downarrow}\; max\, rep$
8. $s_0\, ATu \;\rightarrow\; s_0\; max\, rep$
9. $s_0\, ATd \;\rightarrow\; s_0\; max\, rep$
10. $s_0\, ALu \;\rightarrow\; s_0\; max\, rep$
11. $s_0\, ALd \;\rightarrow\; s_0\; max\, rep$
12. $s_0\, GEu \;\rightarrow\; s_0\; max\, rep$
13. $s_0\, GEd \;\rightarrow\; s_0\; max\, rep$
14. $s_0\, GPu \;\rightarrow\; s_0\; max\, rep$
15. $s_0\, GPd \;\rightarrow\; s_0\; max\, rep$
16. $s_0\, B1o \;\rightarrow\; s_0\; max\, rep$
17. $s_0\, B1i \;\rightarrow\; s_0\; max\, rep$
18. $s_0\, B2o \;\rightarrow\; s_0\; max\, rep$
19. $s_0\, B2i \;\rightarrow\; s_0\; max\, rep$

Rules description:

1: the request is retransmitted to the sources;
2: the cell waits for all sources to answer;
3..4: measured values are relayed to the general controller;
5: cell confirms that all measured values were submitted;
6..7: all commands are submitted to sources;
8..19: cleaning rules.

The symbols *SXu* and *SXd* play two roles:

1. if the cell is in the state *sa*, then the symbol indicating a source state (S) is sent to the general controller;
2. if the cell is in state s_0, then the symbol designating a command forwarded to the each source.

Unlike the external sources, symbols *SXu* and *SXd* have the multiplicity equal with the number of units that the power amount taken can be increased or decreased with.

The symbols' meaning are shown in Table 10.

Symbol	Description
q	Query
a	Answer acknowledge
Sxu	Source can have more load
Sxd	Source decrease load

Table 10. External power sources controller symbols

5.12 External sources (GRID)

The Grid is defined by the installed power (m) and by the actual power given, u. The request from the Grid is about the instantaneous power u, if that is above 0, and the possibility to increase the amount offered if maximum value m has not been reached. The answer can be positive or negative.

The second state occurs when actual increase or decrease commands are received and executed. Unnecessary symbols will be removed from the cell.

Rules for the cell GridX are as follows:

1. $s_0\, q \;\rightarrow\; sqd\, min\, rep$
2. $sqd\, u \;\rightarrow\; squ\, u\, ud\, max\, rep$
3. $sqd \;\rightarrow\; squ\, min\, rep$
4. $squ\, u\, m \;\rightarrow\; squ\, d\, max\, rep$
5. $squ\, m \;\rightarrow\; sqa\, m\, uu\, max\, rep$
6. $sqa\, d \;\rightarrow\; sqa\, u\, m\, max\, rep$
7. $sqa\, uu \;\rightarrow\; sqa\, SXu_{\uparrow}\, max\, rep$
8. $sqa\, ud \;\rightarrow\; sqa\, SXd_{\uparrow}\, max\, rep$
9. $sqa \;\rightarrow\; s_0\, a_{\uparrow}\, min\, rep$
10. $s_0\, SXu \;\rightarrow\; s_0\, u\, max\, rep$
11. $s_0\, SXd\, u \;\rightarrow\; s_0\, max\, rep$

Rules description:

1: request to submit the current state;
2..6: computing the difference between maximum value and current value;
7,8: the cell returns an answer;
9: cell confirms that answers were submitted;
10: the cell increases amount given;
11: the cell decreases power.

The symbols' meaning are shown in Table 11.

Symbol	Description
u	Current consumption
m	Maximum available for consumption
q	Query
a	Answer acknowledge
SXu	Consumption can be increased and the request to increase the used power
SXd	Consumption can be decreased and the request to decrease the used power

Table 11. External source cell symbols

6 Conclusions and future work

In this paper we presented a model for a real-life working system ([6]), we described the use of hyperdag P systems for a feedback-oriented infrastructure that quickly reacts to environment conditions and adapts the parameters accordingly. A detailed description of the system is followed by the complete rule set and transition diagrams, in order to better understand the concept. The model was tested and validated using the ad-hoc built simulator and the results were the ones expected.

Future work involves extending the simulator to accept new types of P systems and development of other models (like network-related algorithms) that can be simulated by using this architecture. Another aspect to be considered is a formal testing of the proposed model, by using techniques like the ones indicated in [5].

Acknowledgements

The authors wish to thank Dr. Raluca Lefticaru for her valuable advice and the SOP-HRD programme of the EU for funding the research. The work of Adrian Zafiu was supported by SOP-HRD grant 89/1.5/S/63700. The work of Cristian Ştefan was supported by SOP-HRD grant 88/1.5/S/52826.

References

1. Michael J. Dinneen, Yun-Bum Kim, and Radu Nicolescu. P systems and the Byzantine agreement. Report CDMTCS-375, Centre for Discrete Mathematics and Theoretical Computer Science, The University of Auckland, Auckland, New Zealand, January 2010.
2. Michael J. Dinneen, Yun-Bum Kim, and Radu Nicolescu. Synchronization in P modules. Report CDMTCS-378, Centre for Discrete Mathematics and Theoretical Computer Science, The University of Auckland, Auckland, New Zealand, February 2010.
3. J. Ellson, E. Gansner, and many others. Graphviz 2.28: Graph visualization software, 2012.
4. Florentin Ipate, Radu Nicolescu, Ionut-Mihai Niculescu, and Cristian Stefan. Synchronization of p systems with simplex channels. *CoRR*, abs/1108.3430, 2011.
5. Raluca Lefticaru, Marian Gheorghe, and Florentin Ipate. An empirical evaluation of p system testing techniques. *Natural Computing*, 10(1):151–165, 2011.
6. P. L. Milea, Adrian Zafiu, M. Dr̨agulinescu, and O. Oltu. Mpp tracking method for pv systems, based on three points prediction algorithm. *UPB Scientific Bulletin*, 72(4):149–160, 2010.
7. Radu Nicolescu, Michael J. Dinneen, and Yun-Bum Kim. Structured modelling with hyperdag P systems: Part A. Report CDMTCS-342, Centre for Discrete Mathematics and Theoretical Computer Science, The University of Auckland, Auckland, New Zealand, December 2008.
8. Radu Nicolescu, Michael J. Dinneen, and Yun-Bum Kim. Structured modelling with hyperdag P systems: Part A. In *Brainstorming Week on Membrane Computing*, pages 85–107, 2009.

9. Radu Nicolescu, Michael J. Dinneen, and Yun-Bum Kim. Structured modelling with hyperdag P systems: Part B. Report CDMTCS-373, Centre for Discrete Mathematics and Theoretical Computer Science, The University of Auckland, Auckland, New Zealand, October 2009.
10. Gheorghe Păun. Computing with membranes. *Journal of Computer and System Sciences*, 61(1):108–143, 2000.
11. Gheorghe Păun, Grzegorz Rozenberg, and Arto Salomaa. *The Oxford Handbook of Membrane Computing.* Oxford University Press, Inc., New York, NY, USA, 2010.

7 Appendix

We present here the complete trace of the P-system evolution, at each step indicating the contents and current states of the cells.

Step	MemConf	Ctrl
0	s0 t23	s0
1	s0 t23 q	sq
2	sqt t23	sq
3	sql t23	sq MT23
4	sqa t23	sq MT23
5	s0 t23	sq MT23 a
6	s0 t23	sq MT23 a
7	s0 t23	sq MT23 a ST20
8	s0 t23	sq MT23 a2 ST20
9	s0 t23	sq MT23 a2 ST20
10	s0 t23	sq MT23 a2 ST20
11	s0 t23	sq MT23 a2 ST20 B1i500
12	s0 t23	sq MT23 a2 ST20 B1i500 CTu GEu788 B1o1000 Su2999
13	s0 t23	sq MT23 a2 ST20 B1i500 CTu GEu788 B1o1000 Su2999 CTd GEd12 B2i1000 Sd
14	s0 t23	sq MT23 a3 ST20 B1i500 CTu GEu788 B1o1000 Su2999 CTd GEd12 B2i1000 Sd CLu GPu793 B2o500
15	s0 t23	sq MT23 a4 ST20 B1i500 CTu GEu788 B1o1000 Su2999 CTd GEd12 B2i1000 Sd CLu GPu793 B2o500 CLd GPd7
16	s0 t23	sq MT23 a6 ST20 B1i500 CTu GEu788 B1o1000 Su2999 CTd GEd12 B2i1000 Sd CLu GPu793 B2o500 CLd GPd7
17	s0 t23	sqc MT23 ST20 B1i500 CTu GEu788 B1o1000 Su2999 CTd GEd12 B2i1000 Sd CLu GPu793 B2o500 CLd GPd7
18	s0 t23	sqc MT3 B1i500 CTu GEu788 B1o1000 Su2999 CTd GEd12 B2i1000 Sd CLu GPu793 B2o500 CLd GPd7
19	s0 t23	sqc MT2 B1i500 GEu788 B1o1000 Su2999 CTd GEd12 B2i1000 Sd CLu GPu793 B2o500 CLd GPd7 ATu
20	s0 t23	sqc B1i500 GEu788 B1o1000 Su2999 CTd GEd12 B2i1000 Sd CLu GPu793 B2o500 CLd GPd7 ATu
21	s0 t23	sqc B1i500 GEu788 B1o1000 Su2999 GEd12 B2i1000 Sd CLu GPu793 B2o500 CLd GPd7 ATu
22	s0 t23	sqc B1i500 GEu788 B1o1000 Su2999 GEd12 B2i1000 Sd GPu793 B2o500 CLd GPd7 ATu
23	s0 t23	sqc B1i500 GEu788 B1o1000 Su2999 GEd12 B2i1000 Sd GPu793 B2o500 GPd7 ATu
24	s0 t23	sqa B1i500 GEu788 B1o1000 Su2999 GEd12 B2i1000 Sd GPu793 B2o500 GPd7 ATu
25	s0 t23	sqa B1i500 GEu788 B1o999 Su2999 GEd12 B2i1000 GPu793 B2o500 GPd7 Atu
26	s0 t23	sqa GEu288 B1o999 Su2999 GEd12 B2i1000 GPu793 B2o500 GPd7 ATu
27	s0 t23	sqa B1o999 Su2999 GEd12 B2i712 GPu793 B2o500 GPd7 ATu
28	s0 t23	sqa B1o999 Su2999 GEd12 GPu81 B2o500 GPd7 ATu
29	s0 t23	sqa B1o999 Su2999 GEd12 GPu80 B2o500 GPd7
30	s0 t23	sqa Su2999 GEd12 GPu80 B2o500 GPd7
31	s0 t23	sqa GEd12 GPu80 B2o500 GPd7
32	s0 t23	sqa GPu80 B2o500 GPd7
33	s0 t23	sqa B2o500 GPd7
34	s0 t23	sqa B2o500
35	s0 t23	sqa

Table 1. System evolution - Part 1

SenzT	CtrlSenz	SenzL	CtrlCons	CtrlGen	CtrlBat
s0 t20	s0	s0	s0	s0	s0
s0 t20	s0 q	s0	s0 q	s0 q	s0 q
s0 t20 q	s0	s0 q	sq	sq	sq
sqt t20	s0	sql	sq	sq	sq
sqa t20	s0 ST20	sqa	sq	sq	sq
s0 t20	s0 ST20 a2	s0	sq	sq	sq
s0 t20	sq ST20 a	s0	sq	sq	sq B1i500 B2i1000
s0 t20	sq a	s0	sq	sq	sq B1i500 B2i1000 B1o1000 B2o500
s0 t20	s0	s0	sq CTu CLu	sq GEu788 GPu793	sq B1i500 B2i1000 B1o1000 B2o500
s0 t20	s0	s0	sq CTu CLu CTd CLd	sq GEu788 GPu793 GEd12 GPd7	sq B1i500 B2i1000 B1o1000 B2o500 a2
s0 t20	s0	s0	sq CTu CLu CTd CLd a2	sq GEu788 GPu793 GEd12 GPd7 a2	sa B1i500 B2i1000 B1o1000 B2o500
s0 t20	s0	s0	sa CTu CLu CTd CLd	sa GEu788 GPu793 GEd12 GPd7	sa B2i1000 B1o1000 B2o500
s0 t20	s0	s0	sa CLu CTd CLd	sa GPu793 GEd12 GPd7	sa B2i1000 B2o500
s0 t20	s0	s0	sa CLu CLd	sa GPu793 GPd7	sa B2o500
s0 t20	s0	s0	sa CLd	sa GPd7	sa
s0 t20	s0	s0	sa	sa	s0
s0 t20	s0	s0	s0	s0	s0
s0 t20	s0	s0	s0	s0	s0
s0 t20	s0	s0	s0	s0	s0
s0 t20	s0	s0	s0	s0	s0
s0 t20	s0	s0	s0	s0	s0
s0 t20	s0	s0	s0	s0	s0
s0 t20	s0	s0	s0	s0	s0
s0 t20	s0	s0	s0	s0	s0
s0 t20	s0	s0	s0	s0	s0
s0 t20	s0	s0	s0 B1o Sd	s0 B1o Sd	s0 B1o Sd
s0 t20	s0	s0	s0 Sd GEu500 B1i500	s0 GEu500 Sd B1i500	s0 B1i500 Sd GEu500
s0 t20	s0	s0	s0 Sd GEu288 B1i500 B2i288	s0 GEu288 Sd B1i500 B2i288	s0 B1i500 B2i288 Sd GEu288
s0 t20	s0	s0	s0 Sd B1i500 B2i1000 GPu712	s0 GPu712 Sd B1i500 B2i1000	s0 B1i500 B2i1000 Sd GPu712
s0 t20	s0	s0	s0 Sd B1i500 B2i1000 GPu ATu	s0 GPu Sd B1i500 B2i1000 ATu	s0 B1i500 B2i1000 Sd GPu ATu
s0 t20	s0	s0	s0 Sd B1i500 B2i1000 GPu	s0 GPu Sd B1i500 B2i1000	s0 B1i500 B2i1000 Sd GPu
s0 t20	s0	s0	s0 Sd B1i500 B2i1000	s0 Sd B1i500 B2i1000	s0 B1i500 B2i1000 Sd
s0 t20	s0	s0	s0 Sd B2i1000	s0 Sd B2i1000	s0 B2i1000 Sd
s0 t20	s0	s0	s0 Sd	s0 Sd	s0 Sd
s0 t20	s0	s0	s0	s0	s0
s0 t20	s0	s0	s0	s0	s0

Table 1. System evolution - Part 2

CtrlGrid	ConsT	ConsL	GenEOL	GenPV
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0 q	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
sq	s0 u10 m100 q	s0 u10 m150 q	s0 u12 m800 q	s0 u7 m800 q
sq	sqd u10 m100	sqd u10 m150	sqd u12 m800	sqd u7 m800
sq	squ u10 m100 ud	squ u10 m150 ud	squ u12 m800 ud12	squ u7 m800 ud7
sq	squ m90 ud d10	squ m140 ud d10	squ m788 ud12 d12	squ m793 ud7 d7
sq	sqa m90 ud d10 uu	sqa m140 ud d10 uu	sqa m788 ud12 d12 uu788	sqa m793 ud7 d7 uu793
sq	sqa u10 m100 ud uu	sqa u10 m150 ud uu	sqa u12 m800 ud12 uu788	sqa u7 m800 ud7 uu793
sq Su2999	sqa u10 m100 ud	sqa u10 m150 ud	sqa u12 m800 ud12	sqa u7 m800 ud7
sq Su2999 Sd	sqa u10 m100	sqa u10 m150	sqa u12 m800	sqa u7 m800
sq Su2999 Sd a	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
sa Su2999 Sd	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
sa Sd	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
sa	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0 Sd B1o	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0 Sd GEu500 B1i500	s0 u10 m100	s0 u10 m150	s0 u12 m800	s0 u7 m800
s0 Sd GEu288 B1i500 B2i288	s0 u10 m100	s0 u10 m150	s0 u12 m800 GEu500	s0 u7 m800 GEu500
s0 Sd B1i500 B2i1000 GPu712	s0 u10 m100	s0 u10 m150	s0 u512 m800 GEu288	s0 u7 m800 GEu288
s0 Sd B1i500 B2i1000 GPu ATu	s0 u10 m100	s0 u10 m150	s0 u800 m800 GPu712	s0 u7 m800 GPu712
s0 Sd B1i500 B2i1000 GPu	s0 u10 m100 ATu	s0 u10 m150 ATu	s0 u800 m800	s0 u719 m800
s0 Sd B1i500 B2i1000	s0 u11 m100	s0 u10 m150	s0 u800 m800 GPu	s0 u719 m800 GPu
s0 Sd B2i1000	s0 u11 m100	s0 u10 m150	s0 u800 m800	s0 u720 m800
s0 Sd	s0 u11 m100	s0 u10 m150	s0 u800 m800	s0 u720 m800
s0	s0 u11 m100	s0 u10 m150	s0 u800 m800	s0 u720 m800
s0	s0 u11 m100	s0 u10 m150	s0 u800 m800	s0 u720 m800

Table 1. System evolution - Part 3

Bat1	Bat2	Grid	Out
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000 q	s0 u2000 m20000 im1000 dm500 q	s0 u m3000 q	
sqd u2000 m5000 im500 dm1000	sqd u2000 m20000 im1000 dm500	sqd u m3000	
sqd m3000 im500 dm1000 d2000	sqd m18000 im1000 dm500 d2000	squ u m3000 ud	
sqi m3000 im500 dm1000 d2000	sqi m18000 im1000 dm500 d2000	squ m2999 ud d	
sqo m3000 im500 dm1000 d2000	sqo m18000 im1000 dm500 d2000	sqa m2999 ud d uu2999	
sqa m3000 im500 dm1000 d2000	sqa m18000 im1000 dm500 d2000	sqa u m3000 ud uu2999	
sqa u2000 m5000 im500 dm1000	sqa u2000 m20000 im1000 dm500	sqa u m3000 ud	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	sqa u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000 B1o	s0 u2000 m20000 im1000 dm500 B1o	s0 u m3000	
s0 u2000 m5000 im500 dm1000 du	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000 du	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000 du	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000 du	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000 du	s0 u2000 m20000 im1000 dm500	s0 u m3000	
s0 u2000 m5000 im500 dm1000 du B1i500	s0 u2000 m20000 im1000 dm500 B1i500	s0 u m3000	
s0 u2000 m5000 im500 dm1000 du B2i1000 iu500	s0 u2000 m20000 im1000 dm500 B2i1000	s0 u m3000	
s0 u2000 m5000 im500 dm1000 B2i1000 iu499	s0 u2000 m20000 im1000 dm500 iu1000	s0 u m3000 Sd	
s0 u2000 m5000 im500 dm1000 iu499	s0 u2000 m20000 im1000 dm500 iu1000	s0 m3000	

Table 1. System evolution - Part 4