EvoNILM - Evolutionary Appliance Detection for Miscellaneous Household Appliances

Dominik Egarter, Wilfried Elmenreich
Institute of Networked and Embedded Systems
Alpen-Adria University Klagenfurt
{dominik.egarter, wilfried.elmenreich}@aau.at

Abstract — *To improve the energy awareness of consumers, it is necessary to provide them with information about their energy demand, not just on the household level. Non-intrusive load monitoring (NILM) gives the consumer the opportunity to disaggregate their consumed power on the appliance level. The consumer is provided with information about the energy demand of each individual appliances. In this paper we present an evolutionary optimization algorithm, applicable to NILM purposes. It can be used to detect appliances with a probabilistic power demand model. We show that the detection performance of the evolutionary algorithm can be improved if the single population approach of the evolutionary algorithm is replaced by a parallel population approach with individual exchange and by the introduction of application-oriented preprocessing and mutation methods. The proposed algorithm is tested with Matlab simulations and is evaluated according to the fitness reached and detection probability of the algorithm.*

1 Introduction

The smart grid adds information and telecommunication technologies to our current electrical grid to improve the existing grid in order to create a more efficient, reliable and sustainable grid. With the use of additional information about consumed power and current costs of energy, the consumer gets the opportunity to develop and to strengthen their energy awareness, where the incentive for the consumer to be energy-aware is twofold: Firstly, the consumer gets chance to save money by reducing their energy consumption and secondly, the environmental pollution is decreased as a result of the consumer's consciousness about their energy consumption [2].

A possible information channel to improve the energy awareness of consumers is by giving them the knowledge of the power consumption of appliances within the household [2]. By disaggregating the total power consumption to an individual appliance level, the consumer gets the opportunity to develop and improve steps to reduce their energy consumption by recognizing the power demands of individual appliances in real-time. This can lead to an energy saving of 12%. In contrast, in [11] it is mentioned that the knowledge of energy can also be counterproductive or neglectable to energy saving.

However, non-intrusive load monitoring (NILM), introduced by G.Hart [8], deals with the possibility of disaggregating the individual appliance power consumption from the total consumed power of a household. The basis of NILM is it that different appliances have different power consumption characteristics. In [13], the four general appliance families are identified as: permanent consuming appliances, on/off appliances, multi-state appliances and continuous consuming appliances. Beside this knowledge, different solving approaches, such as pattern recognition [10, 4] or optimization techniques [12], are applied to the problem of identifying and disaggregating appliances according to their power characteristics. In general, the structure of a NILM system can be splitted up into a data acquisition layer, an event detection layer [5], a feature extraction layer [3] and a classification layer [10].

In this paper we address the problem of classification of power profiles through the use of a heuristic optimization technique. More specifically, we are matching saved energy patterns of appliances into the total power load. We use the evolutionary algorithm as an optimization technique to minimize the error between evolved power consumption and the total power load. The evolutionary algorithm is advantageous for search and pattern matching tasks, because of its ability to produce solutions using only a simple problem description and because of introducing parallel search for potential solution establishments.

Our approach is based on the work in [7], where it is shown that the evolutionary algorithm is a simple and feasible solution for NILM use. The purpose of the algorithm is to find a composition of saved power profiles, place this composition in time and try to minimize the error between the evolved power load and the given total power load. The algorithm is based on an abstract and simplified definition of the disaggregating problem. It uses an appliance model, where on/off appliances are described by their power magnitude and time usage. This leads to more precise description of the appliance model than by using either the power magnitude or the time duration and accordingly[1], leads to an decrease of the detection search space. Appliance use can only occur once without a repetitive or periodic behavior and cannot consume power constantly. This, and further restrictions, which are described in Section 2 in greater de-

[1] appliance can have same power magnitude or time duration

tail, are overcome in this paper by introducing several pre-processing and mutation operations. With the presented algorithm, miscellaneous appliances with permanent power consumption, simple on/off appliances and appliances which run in a repeating or periodic manner can be detected and identified. Beyond that, we introduced in our evolutionary algorithm a parallel evolution (PE) concept in contrast to the simple evolutionary (SE) concept in [7]. This leads to a performance improvement of the algorithm for both the reached fitness and detection probability. A further important step to make the algorithm more realistic to real-world appliances and usages is to introduce randomly varying time durations for an appliance as well as randomly varying power magnitudes. Therefore, we used normal and gamma distributed usage times of appliances and uniformly distributed power magnitude. This assumption changes the simple approach of constant pattern matching to a probabilistic pattern matching approach.

The remainder of this paper is organized as follows: in Section 2 the general treated optimization problem is explained. In Section 3 the evolutionary optimization approach with its algorithm characteristics and functioning is presented, followed by Section 4, where we describe the used evaluation settings. To verify the proposed algorithm we introduced Section 5 and 6 present fitness trends of the evolutionary algorithm and the reached detection probabilities. Finally, the paper is concluded and future work is discussed in Section 7.

2 Problem Statement

Household appliances consume power in a characteristic way, which makes it possible to distinguish between different appliance types. The technique of non-intrusive load monitoring (NILM) uses this fact to try and determine which type of appliance was used by knowing only the total power consumed by household and the power characteristics of an appliance type. In detail, NILM tries to find out, which in a database stored power profiles $P_i(t)$ occurred in a given total power load $P(t)$. Thus, the total power load, $P(t)$, depends on which appliances and correspondingly, which power profiles, P_i, are used as well as the appliances switched on at a given time. The switching-on event of an appliance is defined as ts_j, where j counts the on-switching events of an appliance. The general appliance power profile $P_i(t)$ is defined as:

$$P_i(t) = p_i \cdot t_i, \tag{1}$$

where p_i is the appliance power and t_i is the usage duration of a specific appliance. By modeling the switching behavior as a state switching vector a_i for every used appliance, a binary vector is constructed by $a_i(t) = 1$ from $t = ts_j$ to $t = ts_j + t_i$ and 0 elsewhere. The total power load $P(t)$ can be presented by:

$$P(t) = \sum_{i=1}^{n} p_i \cdot a_i(t), \tag{2}$$

where n^2 represent the number of appliances used. The aim of NILM is it to approximate the total power load, $P(t)$, with a sum of superimposed appliance power profiles $P_i(t)$, which results in the following optimization problem:

$$e(t) = \arg\min \left| P(t) - \sum_{i=1}^{n} p_i \cdot a_i(t) \right|. \tag{3}$$

As mentioned in [8], this optimization problem is a NP-hard problem, which is computationally intractable. We solve this optimization problem by using an evolutionary algorithm, which is based on the work in [7]. The approach begins by finding the starting events of appliances by edge detection, followed by the placement of a random composition of appliance power profiles (which are stored in a database) at the computed starting events in such a way that the randomly evolved composition of power profiles approximates the measured total power load with maximum fitness. The basic functioning of this evolutionary NILM technique is shown in Figure 1. In [7], it was shown that the presented evolutionary algorithm is feasible for NILM purposes with the following restrictions:

- No permanent, periodic or repetitively consuming appliances.
- The algorithm is highly dependent on power profiles not stored in search database.
- The appliances are modelled with constant power and constant usage times.

In detail, the abstraction of appliances with constant power magnitude, constant time duration and with unique appearance in the observation window is an unrealistic mapping of reality. Thus, we introduce the appliance definition so that appliances can vary their power magnitude as well as their usage duration. Specifically, for every appliance the normal time of use varies from user to user and from usage to usage. To simulate this behavior and also correspondingly, to overcome this obstacle, we need to know which distribution matches best the on-duration distribution of appliances. In [9], it is claimed that the gamma distribution approaches the on-duration distribution of appliances best. In our approach we assume that all appliances are behave independently of each other and in addition, we used additional to the

[2] In this case the number of stored and for the evolutionary algorithm used appliances is the same

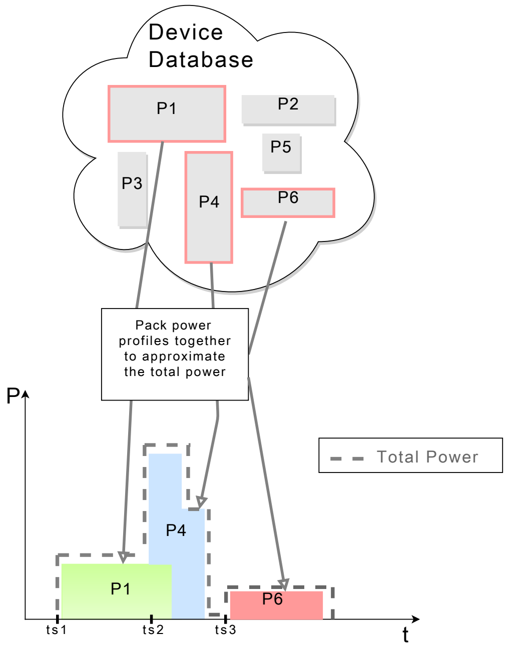


Figure 1: Basic principle of the ON/OFF time genome appliance detection. Find a composition of saved power profiles P_i, place it at the switching times t_s and try to minimize the error between the evolved power load and the given total power load.

gamma distributed on-duration a normal distributed on-duration. However, an appliance is manufactured with the assumption, that it will consume a predetermined amount of energy. The energy demand of an appliance should not differ from manufactured appliance to appliance. This assumption is more or less theoretical. In practise, the energy demand varies from appliance to appliance. Accordingly, a realistic appliance model should not have a constant power, because the power of any appliance varies within a special range [6]. An illustration for the varying usage and power is shown in Figure 2. Beside the variation of time and power, we also stressed the fact of repeating and periodic appearance of appliance, permanent power consuming appliances and appliances not saved in a database. To show that also a simple optimization approach can solve the problem of aggregated power profiles for miscellaneous appliances, we introduce an evolutionary optimization approach named *evoNILM*. The general scheme of the proposed algorithm is shown in Figure 3 and is described in detail in the following sections.

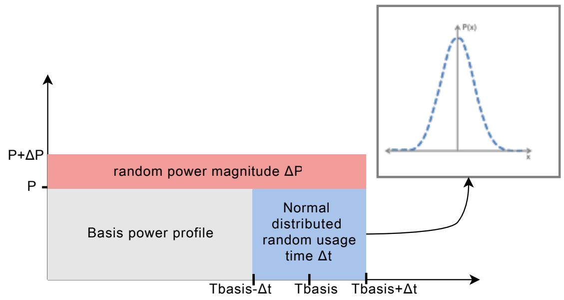


Figure 2: Each appliance power profile is defined by its power magnitude P and its base usage time ,t_{basis}. Due to the fact that usage times are not constant in reality, we vary the usage time by a normal distributed time variable ,Δt, which is added/subtracted to/from the basis/ mean base usage time, t_{basis} or a gamma distributed time $(\Gamma(\alpha = 2, \beta = 50))$ variable, which is added to the base usage time t_{basis}. Beside the usage, also the power is diversified as we vary the power P with an additional uniform distributed random variable, ΔP.

3 THE EVONILM ALGORITHM

The evoNILM algorithm can be divided into two main stages: a pre-processing stage and the evolutionary algorithm stage. The pre-processing stage is responsible mainly for identifying predefined characteristics of the total power load e.g., if repeating, periodic or permanent power signals are occurring in the total power load. The evolutionary algorithm stage is the main stage which performs the optimization between the generated power signal of the evolved composition of appliances and the total power load.

3.1 PRE-PROCESSING STAGE

The first step of the pre-processing stages is to check if the total power load contains periodic or repeating signals [3]. This is necessary because the knowledge of the appearance of appliances with repetitive and periodic characteristics is later used in the two new introduced mutation methods. These two mutation methods are used to improve the ability to detect appliances with repetitive and periodic running properties.
The second step of the pre-processing stage is responsible for verifying if the total power load contains power loads which are not known and thus are not saved in the database. If these additional unknown loads, were also be considered during the evolutionary optimization process, the detection performance would decrease [7]. To overcome this problem, we introduce checking for unknown power loads. This occurs by processing the positive and

[3] A detailed description how the detection of repeating and periodic appliance can be found in 3.2.

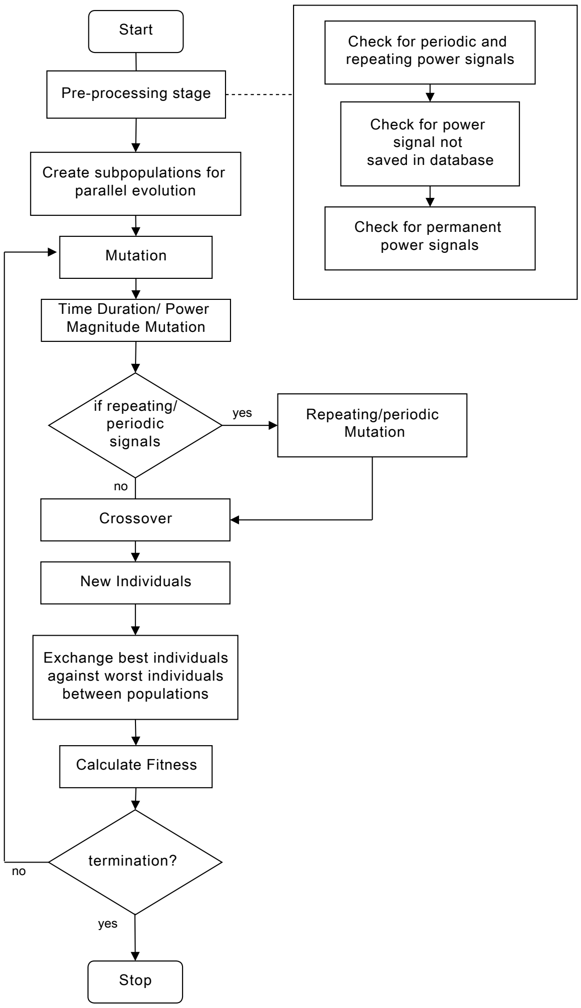


Figure 3: General scheme of the evoNILM algorithm as a flowchart

negative edges that occur in the total power load, followed by a check to determine whether the detected positive and negative edges correspond to any in the database saved power magnitudes. If positive and negative edges appear in the total power load, which are not saved in the database, the algorithm takes these edges and according power value, p_d and usage time, t_d between positive and negative edges and withdraws the edge values, p_d from the total power load in the processed usage times, t_d. The result of this pre-processing step is that the algorithm uses a total power load of known power profiles and accordingly, becomes independent of additional unknown power profiles. It tries to reduce the search space of the algorithm with the aim of simplifying the optimization problem.
In addition to appliances which consume their power by alternately switching on and off, there exist appliances which consume power permanently. This behavior of permanent power consumption makes the optimization problem very complicated and intractable. Thus, we introduced the third and final pre-processing stage with the aim of detection these permanently-occurring signals and removing them from the observed total power load [4]. In detail, we perform an edge detection to find the first rising edge and compare the computed edge value with the real power value of the total power load at the position of the first rising edge. If these two values are different, this indicates that a permanent power consumption is present in the total power load. To ensure this assumption, we calculate the difference between the edge value and the total power load value and remove this value from the whole total power load. If the produced difference signal is at any point of the observation window greater than or equal to zero, this indicates that a permanent signal is present during the whole observation window. As a consequence we withdraw the computed permanent power from the total power load.

3.2 EVOLUTIONARY ALGORITHM

The main task of the evoNILM algorithm is the evolutionary algorithm, which aggregates a random composition of stored power profiles $P_i(t)$ to approximate the total measured power profile $P(t)$ in the best way like in [7]. For this we use the following fitness function F:

$$F = -\left| P(t) - \sum_{i=1}^{Nb} p_{d_i} \cdot a_i(t) \right|, \tag{4}$$

where Nb is the number of appliances used and d_i is the genome index[5], which indicates which appliance according to its database index, d_i (Figure 4) should be used for each state switching vector, $a_i(t)$. Thus, the closer the fitness is to zero the fitter a desired population is. In the proposed evolutionary approach we used the common evolutionary operators of uniformly mutation, single-point crossover, elite selection and newly generated individuals per generation. To overcome the restriction of the simple optimization approach of [7] we introduced the following four mutation operators.

3.2.1 Time-Duration Mutation

Each appliance has its base usage time and can randomly change the real usage time by a normal or gamma distributed additional time variable, Δt. To give the algorithm the ability to overcome the lack of knowledge according to the additional random usage time Δt, we

[4]It is assumed that all related appliance events are occurring in the observation window. All on and corresponding off events are in the observation window

[5]The appliance power profiles $P_i(t)$ are stored in a database and is accessible by its database index

introduce a mutation method for our evolutionary algorithm, which creates normal or gamma distributed usage times. An illustration of the normal and gamma distributed usage time is shown in Figure 2. Depending on the mutation rate, we randomly extend/reduce the usage time by a normal or gamma distributed time variable to approximate the real time of usage. The varying random time Δt is only saved in one specific population. Therefore, for every new generation the random time of the previous generation is used and updated according to the time-duration mutation.

3.2.2 Power Magnitude Mutation

We introduced a probabilistic varying power magnitude for our appliance model by changing the power magnitude according to a predefined tolerance scheme. The power magnitude mutation adds to every stored power magnitude in the database an additional power value, ΔP, which is randomly [6] generated within a predefined tolerance percentage. The introduction of this mutation method leads to an increase in the search diversity and leads to an improvement in terms of fitness and detection evolution. An illustration of the base power magnitude and the additional power magnitude ΔP is presented in Figure 2.

3.2.3 Repeating-Signal Mutation

According to the common usage of appliances, it very often happens that appliances are frequently and repeatedly used during a given time window. Thus, we introduced the repeating-signal mutation which helps to find and to introduce repeating appliances into the evolution process. More precisely, this mutation method tries to find rising edges and stores the magnitudes of the found edges. If the same magnitudes appear several times, the algorithm saves the corresponding starting events for the repetitive magnitudes, randomly creates a new appliance index and inserts this index at the saved starting events into the current executed population. This process means that the same appliance index is used at the stored repetitive starting events. As mentioned in 3.1, we included at the beginning of the optimization process a pre-processing stage, which is responsible for finding out, if repeating signals appear in the total power load and thus, whether the repeating-signal mutation should be used or not.

3.2.4 Periodic-Signal Mutation

Nowadays, many appliances behave periodically. They turn on and off at almost equally spaced intervals. Accordingly, the periodic-signal mutation tries to detect these periodic occurring events and therefore, introduces the same appliance index at equally spaced intervals. More specifically, the algorithm tries to detect all rising edges and calculates the time difference Δt_p between each rising edge. These values of Δt_p's are then compared between each other, if one or more time differences are repeated. If a Δt_p is repeated more than twice, the associated rising edges and starting events are saved and a randomly generated appliance index is inserted at the saved starting events. As in the case of repetitive signal, we also try to detect periodic signal at the pre-processing stage to determine whether the periodic-signal mutation should be used or not.

By introducing these four mutation methods the evolutionary optimization algorithm can be improved and can overcome many of the restrictions seen in [7]. In general, the evolutionary algorithm can achieve a good optimization solution in reasonable time but if we use the evolutionary algorithm to solve more complicated problems with a large search space, such as the load disaggregation problem with many different appliances, the time taken to achieve a satisfying and reasonable solution increases. To surmount this problem, we attached parallel evolution to our algorithm. More precisely, we used a multiple-population evolutionary approach, where r_i populations are evolved separately as single evolutions and exchange the best individuals between each other at predetermined generation intervals. In our approach we used a ring-topology of populations in such a manner that each population acts as a single evolution and exchange its best individuals, bi to the right-hand neighbor at predetermined time intervals. Thus, the bi worst individuals are replaced by the bi best individuals of the left neighbor. This behavior is propagated over the whole ring and leads to an improvement of the fitness development and to an increase in speed up of the algorithm according to the new introduced population diversity. Finally, the parallel evolution [7] does not deliver just one population solution, it provides r solutions, which correspond to the size of the population ring topology.

4 EVALUATION SETTINGS

The evaluation of the proposed evolutionary algorithm is based on MATLAB simulations, in which we simulated 10 randomly generated households with Nb appliances. Each generated house has its own appliances with their power profiles, Pi, which are randomly aggregated to establish the total power load $P(t)$ for every simulation run. Appliances are randomly selected from a database

[6] Uniformly distributed

[7] For every population we used different evolutionary operators with the exception of the the time-duration mutation and the standard evolutionary operators such as crossover, uniform mutation and new individuals. If the used number of parallel populations is 4, we used half of them with repeating mutation and the other half with periodic mutation

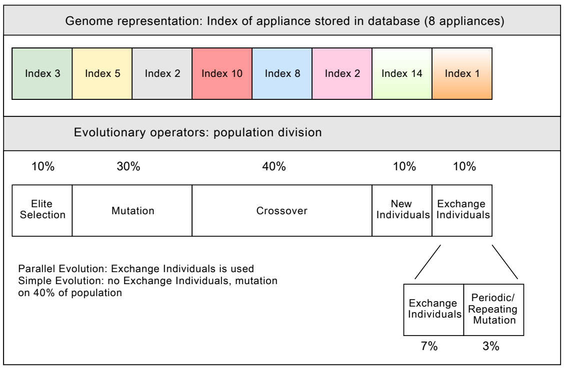


Figure 4: Genome representation uses real values according to the appliance index stored in the database. Furthermore, the division of evolutionary operators such as mutation, crossover, etc., are shown.

of size $db = 25$ and are defined by their power magnitude, p_i and their usage time t_i. Thus, the appliance power profiles can be seen as rectangular power signals, which can be further varied in time and power magnitude. The base time t_{basis}, which corresponds to t_i of P_i, is varied by Δt (Figure 2) and the power magnitude p_i is uniformly increased by $5 - 20\%$ of the basis power consumption p_i. However, we used an observation window of 7200 data points for our simulation, which can be seen as a time window of 2 hours for a day in second time resolution. We also randomly generate the used starting times t_s to place the randomly chosen appliances in the observation window and to create a total power load $P(t)$. Moreover, we repeat this random generation of starting points for each household 10 times to simulate 10 independent simulation runs for each generated house and household appliances. The composition of appliances stays the same and only the starting times are changing for each of these 10 simulation runs per household. Nevertheless, to create a properly functioning evolutionary algorithm, the parameters of the algorithm have to be fixed. In Table 1, the general parameters of the evolutionary algorithm, such as the mutation rate and the specific parameters of our proposed evolutionary algorithm are defined. The parameters were determined empirically. In Figure 4 the genome representation and the division of the evolutionary operators in one generation are shown. For the evaluation of our algorithm we use the metric of the fitness behavior during the detection process with the aim of seeing how accurate the appliance detection is in the sense of fitness reached and how fast (in generations) the total process is to find a reasonable detection result. In contrast to the fitness metric we also use a detection metric, where we evaluate the probability of detected appliances in the mean of all simulation runs, which corresponds to a true positive probability. It compares the probability of appliances correctly detected to total number of actual appliances [1]. In [7] simple evolution (SE) was used. In this paper we improved the algorithm behavior by a multiple/parallel population (PE) approach. We used parallel evolution with r_i independent sub-populations and with individual exchange to improve the detection and fitness behavior of the algorithm. In Figure 5 we show the different performance results between simple and parallel evolution under varying wanted appliances. In detail, we used the appliance number $Nb = [5, 9, 12]$ and $r_i = 4$ parallel evolutions. It can be seen that the evolution process of the parallel evolution improves its fitness trend if the number of appliances is increased. This is also visible in Table 2, where the detection probabilities for different test cases are listed. For example, $Nb = 9$ with simple evolution a detection probability of 79% can be reached. This can be improved by the parallel evolution to a detection probability of 92%. Beyond that, to evaluate our new introduced mutation methods to improve the evolutionary NILM algorithm we defined the test scenarios *time and power magnitude varying appliance evolution* and *repeating and periodic appliance evolution*. In the

Description	Value range
Mutation rate :	
-Evolutionary operator mutation	0.1
-Repeating-signal mutation	0.1
-Periodic-signal mutation	0.1
-Time duration & power magnitude mutation	0.1
Number of generations	100
Number of individuals	500

Table 1: Parameters used for the evolutionary algorithm

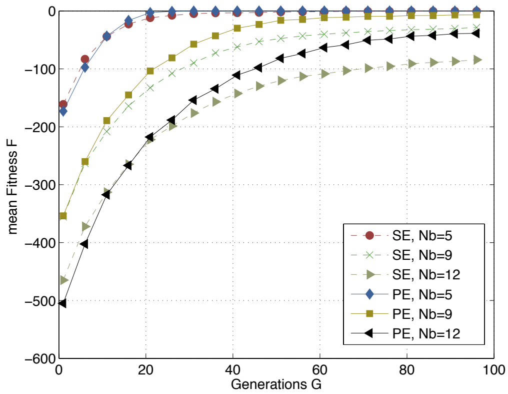


Figure 5: Fitness trends for simple (SE) and parallel evolution (PE) with varying instance number r_i of parallel evolutions and $Nb = 7$

Simulation case	SE P_{det}	PE P_{det}
$NB = 5$	**99%**	**100%**
$NB = 9$	79%	92%
$NB = 12$	55%	75%

Table 2: Detection probability for simple (SE) and parallel evolution (PE) with varying number of appliances $Nb = [5, 7, 9, 12]$

following sections we are describing and outline these scenarios and evaluate the performance of the evolutionary algorithm according to the achieved detection results.

5 Evolution with varying usage times and power magnitude

To vary the usage time of appliances we vary the base usage time of a saved appliance dependent on a normal or gamma distributed time variable Δt. For the normal distributed case we use the base time t_{basis} as mean value with a variance of $75W$ as the varying time variable Δt and in the gamma distributed case, we increase the base time t_{basis} by $\Delta t = \Gamma(2, 50)$. For the simulations we presume to know on which distribution the time variation is based. Accordingly, with the knowledge of varying time instances we introduced a new mutation method which also tries to improve the algorithm performance also if the exact time of usage for a wanted appliance is not known. In Figure 6(a) we show the fitness trend for the cases with simple or parallel evolution with time varying usage times and with/without time duration mutation. We used normal and gamma distributed time durations. It can easily be seen that the algorithm is heavily dependent on the influence of varying usage times and also on the usage time distribution. Thus, if the time duration is normal distributed, the new mutation method does not improve the detection behavior. Only the fitness is increasing. The detection probability is even worse than without time-duration mutation. In contrast, with a gamma distributed usage time the detection behavior can be improved from 46% to 66%. A futher improvement is also possible with the use of parallel evolution instead of simple evolution. As mentioned in the previous section, appliances do not only differ in usage time, they can also differ in their power magnitude. Accordingly, we vary the power magnitude randomly with an additional power value Δp of 5% to 20% of the basis saved power magnitude p_i. To overcome and to improve the detection performance we introduced the power magnitude mutation and the enhancement can be seen in Figure 6(b) and Table 4.

Simulation cases	P_{det}
SE, normal distr., *no* time-duration mutation	63%
SE, normal distr., *with* time-duration mutation	60%
SE, gamma distr., *no* time-duration mutation	46%
SE, gamma distr., *with* time-duration mutation	66%
PE, gamma distr., *no* time-duration mutation	56%
PE, gamma distr., *with* time-duration mutation	**76%**

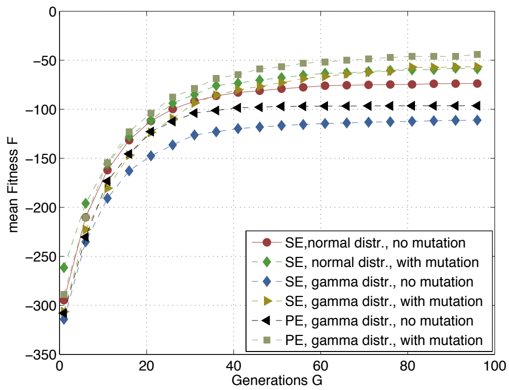
Table 3: Detection probability with/without time duration mutation for simple (SE) and parallel evolution (PE)

Simulation cases	P_{det}
SE, *no* power magnitude mutation	65%
SE, *with* power magnitude mutation	**72%**
PE, *no* power magnitude mutation	75%
PE, *with* power magnitude mutation	**89%**

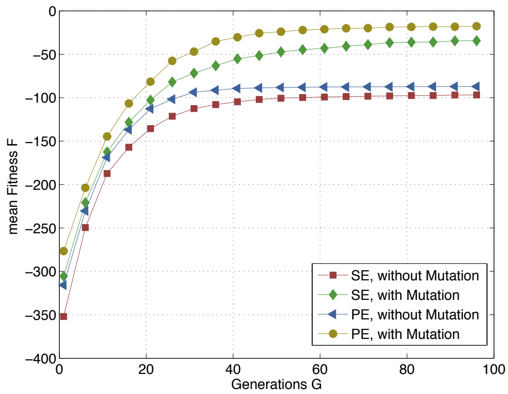
Table 4: Detection probability with/without power magnitude mutation for simple (SE) and parallel evolution (PE)

6 Repeating and Periodic Appliance Environment

In our households many appliances are used repeatedly and periodically. Therefore, we introduced two mutation methods to improve the detection abilities of the evolutionary algorithm for these appliance types. In contrast to before, we now include repeating and periodic appliances in the total power load $P(t)$. In every simulation run, we place the same appliance type into the observation window several times. Appliances, which are repeatedly used, are usually heating appliances, which consume power according to the hysteresis curve of the thermostat of heating elements. Therefore, it is imaginable that an appliance is on longer at the starting time than during subsequent operating times. However, in the case of periodic appliances we place the same appliance type with constant power magnitude p_i and time duration t_i into the simulated total power load. The time difference from one starting event to the next starting event of a desired appliance is equally spaced and the appliance is therefore periodic. For every simulation run we used $Nb = 7$, which corresponds to the used number of start-



(a) Fitness trends for normal and gamma distributed usage times for appliance with/
without time duration mutation for simple (SE) and parallel evolution (PE)



(b) Fitness trends with/without power magnitude mutation for simple (SE) and parallel evolution (PE)

Figure 6: Fitness Evaluation for time varying and power magnitude varying appliance model

ing events. To clarify the effect of the newly introduced mutation methods for repeated and periodic use appliances, we used the single population approach instead of the parallel population approach. For the repetitive

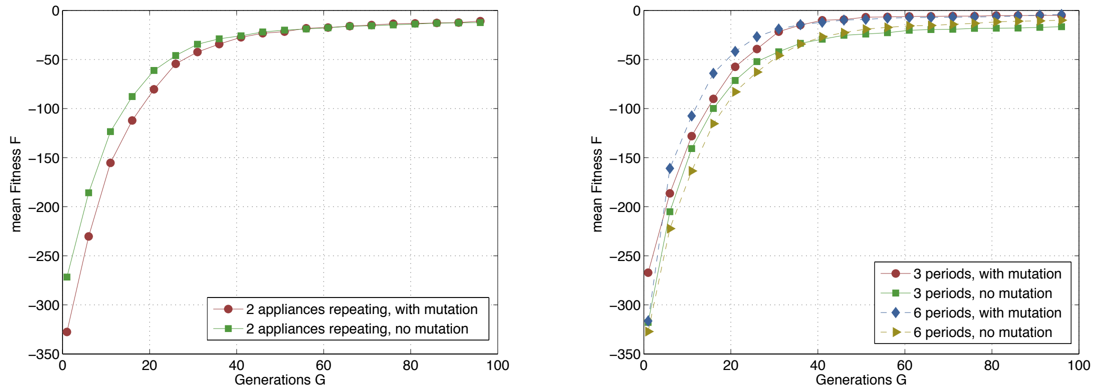
Simulation cases	P_{det}
2 appliances repeating, *with* repeating mutation	**88%**
2 appliances repeating, *no* repeating mutation	85%
appliance with 3 periods, *with* periodic mutation	**96%**
appliance with 3 periods, *no* periodic mutation	86%
appliance with 6 periods, *with* periodic mutation	**94%**
appliance with 6 periods, *no* periodic mutation	92%

Table 5: Detection probability with/without repeating or periodic mutation for simple evolution (SE)

appliances scenario we choose to have 2 random power profiles, which were placed into the desired observation window twice. We used our evolutionary algorithm with and without repeating-signal mutation in Figure 7(a) and in Table 5. It can be seen that the fitness trend leads to a faster result and the detection probability is improved. To evaluate the periodic behavior of evoNILM we used one randomly generated appliances with 3 or 5 periods. Just as in the case of repeating appliances, we used $Nb = 7$. We evaluate our algorithm according the achieved fitness in Figure 7(b) and the detection probability in Table 5 with and without the new mutation method. In the case of 3 periods it can be seen that our proposed mutation method yielded to an improvement from the fitness reached as well as for the detection probability. This effect is also visible for the case of 6 periods. With the introduction of these two mutation methods the algorithm can be improved by adding information about the search space (pre-processing stage) and by adding possible solutions (repeating mutation method) to the algorithm.

7 Conclusion and Future Work

In this paper we presented an evolutionary approach to detect miscellaneous appliances based on their power characteristics. The proposed algorithm is feasible for detecting and identifying appliances, which behave permanently, repetitively and periodically. The appliance can be described as on/off appliances with randomly distributed power demand and usage times. We showed that the detection performance can be increased by introducing new pre-processing steps and mutation methods and that by changing the single population approach of the evolutionary algorithm to a multiple-population approach with individual exchange the performance of the algorithm can be increased. With parallel evolution the fitness reached by the evolutionary algorithm and the detection probability can improved in the sense that a wide range of appliances can be detected with a higher detection probability. In future we wish to apply our algorithm to real power profiles of appliances, where the algorithm performance can be evaluated in reality instead of a simulation-based environment. Further, we plan to compare the evolutionary optimization approach with other heuristic optimization techniques like particle



(a) Fitness trends of simple evolution (SE) for 2 repeating appliance with/without the use of repeating mutation

(b) Fitness trends of simple evolution (SE) for appliances with 3 and 6 periods and with/without the use of periodic mutation

Figure 7: Fitness Evaluation for repeating and periodic appliances

swarm optimization.

REFERENCES

[1] K. Anderson, M. Berges, A. Ocneanu, D. Benitez, and J. M. F. Moura. Event detection for non intrusive load monitoring. In *Proceedings of the 38th Annual Conference on IEEE Industrial Electronics Society (IECON)*, 2012.

[2] K. C. Armel, A. Gupta, G. Shrimali, and A. Albert. Is disaggregation the holy grail of energy efficiency? the case of electricity. *Energy Policy*, 52(0):213 – 234, 2013.

[3] H.-H. Chang, P.-C. Chien, L.-S. Lin, and N. Chen. Feature extraction of non-intrusive load-monitoring system using genetic algorithm in smart meters. In *Proceedings of the IEEE 8th International Conference on e-Business Engineering (ICEBE)*, 2011.

[4] H.-H. Chang, C.-L. Lin, and J.-K. Lee. Load identification in nonintrusive load monitoring using steady-state and turn-on transient energy algorithms. In *Proceedings of the 14th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, pages 27–32, 2010.

[5] R. Cox, S. Leeb, S. Shaw, and L. Norford. Transient event detection for nonintrusive load monitoring and demand side management using voltage distortion. In *Proceedings of the 21st IEEE Applied Power Electronics Conference and Exposition (APEC)*, 2006.

[6] M. Dong, P. C. M. Meira, W. Xu, and C. Y. Chung. Non-intrusive signature extraction for major residential loads. *IEEE Transactions on Smart Grid*, PP(99):1 –10, 2013.

[7] D. Egarter, A. Sobe, and W. Elmenreich. Evolving non-intrusive load monitoring. In *Proceedings of the Applications of Evolutionary Computation*, volume 7835 of *Lecture Notes in Computer Science*, pages 182–191, 2013.

[8] G. Hart. Nonintrusive appliance load monitoring. *Proceedings of the IEEE*, 80(12):1870–1891, 1992.

[9] H. Kim, M. Marwah, M. F. Arlitt, G. Lyon, and J. Han. Unsupervised Disaggregation of Low Frequency Power Measurements. In *Proceedings of the 11th SIAM International Conference on Data Mining*, pages 747–758, 2011.

[10] J. Liang, S. K. K. Ng, G. Kendall, and J. W. M. Cheng. Load signature study part i: Basic concept, structure, and methodology. *IEEE Transactions on Power Delivery*, 25(2):551–560, 2010.

[11] F. Mattern, T. Staake, and M. Weiss. ICT for green: how computers can help us to conserve energy. In *Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*, e-Energy '10, pages 1–10, 2010.

[12] K. Suzuki, S. Inagaki, T. Suzuki, H. Nakamura, and K. Ito. Nonintrusive appliance load monitoring based on integer programming. In *Proceedings of the SICE Annual Conference*, pages 2742–2747, 2008.

[13] M. Zeifman and K. Roth. Nonintrusive appliance load monitoring: Review and outlook. *IEEE Transactions on Consumer Electronics*, 57(1):76 –84, 2011.