Extremal Optimisation Approach to Component Placement in Blood Analysis Equipment

Magnus Sethson[1]

[1]Div. of Fluid and Mechatronic Systems (FLUMES),
Dept. of Management and Engineering (IEI),
Linköping University, Sweden

May 31, 2021

Abstract

This reports present an initial study on generative mechatronic design of equipment for blood analysis where the samples and chemicals are forwarded in thin single millimeter vessels. The system of vessels in the equipment transfers the fluids to different stations where chemical reactions and studies are performed. One of the stations is an optical inspection that requires controllable lighting conditions using an array of LEDs of different types.

The focus is on the generative design of the placement and configuration of the LEDs. The placement of the LEDs has been taken as a studying case for the method of Extremal Optimisation (EO) approach to mechatronic design. This method forms an opposing strategy to methods like genetic algorithms and simulated annealing. This is because it discriminate the individual parts or components of the configuration that underperform in a particular aspect instead of the more classical strategy of favouring good configurations from global measures. The presented study also relates to the class of many-objective optimisation methods (MaOP) and originates from the concept of self-organised criticality (SOC). The characteristics of avalanche barrier crossings in the parameter search space is inherited from such systems.

The test case used for the evaluation places occupying circles onto a quarter ring domain representing LEDs and circuit board. The fluid vessels are represented by lit up small domains that are also approximated by a circular disc. Some conclusion upon the methods capability to form a valid solution are made. A framework for describing a set of local flaw-improvement rules, called **D2FI** is introduced.

1 Background

This article reports the current state of an ongoing research project at Linköping University, where *Biomedical and Clinical Sciences* and *Management and Engineering* departments collaborate within the project CLOTRETRACT. The work presented here relates to some engineering aspects of defining and building mechatronic parts of a prototype machine for a new type of automated blood analysis. Most of the presented study relates to the Extremal Optimisation (EO) method and is carried out by the author at the Fluid and Mechatronic Systems division at Linköping University.

The general layout of the prototype and experimental machine for the blood analysis is shown in figure 1. Its overall structure goes from a particular camera in the base to the sample holder on the top. In the middle, there is a circuit board (PCB) holding several LEDs (Light Emitting Diode) for illumination of the six sample chambers at the top. The layout of this LED PCB is the focus of this article. Its configuration both in terms of shape, LED population, and component placement is delicate and affects the general performance of the automated analysis process. An external computer controls both the camera and LED synchronisation. The LED configuration act as an illumination source with angular characteristics to the optical axis through the machine. .

2 Introduction

The layout of PCBs is called auto-routing, most often solely related to generating the copper paths on different layers of the PCB so that the mounted components on the surfaces of the PCB are connected correctly. The routing process may occur with conditional requirements regarding the maximum current of the trace, high-frequency crosstalk, or other geometrical hinders. Most modern software packages (EDA) have auto-routing with conditions built-in today.

In this work, most of the component placement have stipulated boundary conditions set by external factors, and therefore the software available is not suitable for an automated design process of the PCB for this type of machine. By combining the external conditions into the automated design process, one needs to include geometry conditions from other levels than the PCB surface along the optical axis.

First studies on this indicated that setting up a general optimisation problem for such a design process was not achievable due to the difficulties of finding a reasonable starting condition. Both genetic algorithms and simulated annealing were tested. The problem proved to include some barriers that were very difficult or costly to cross. The search process was often stuck, or the progress was plodding, indicating that the approach used was not the correct one.

The problem at hand has the characteristics of both varying topology and combinatorics. The number of LEDs on the PCB is not given beforehand. Instead, a maximum surface area is provided and should include a particular mix of LEDs. The LEDs come in different sizes and have different emission angular openings and therefore need to be placed at a certain distance from the affected test chamber. The ranges also vary with the type of LED. The illumination from some LEDs is also believed to be more critical

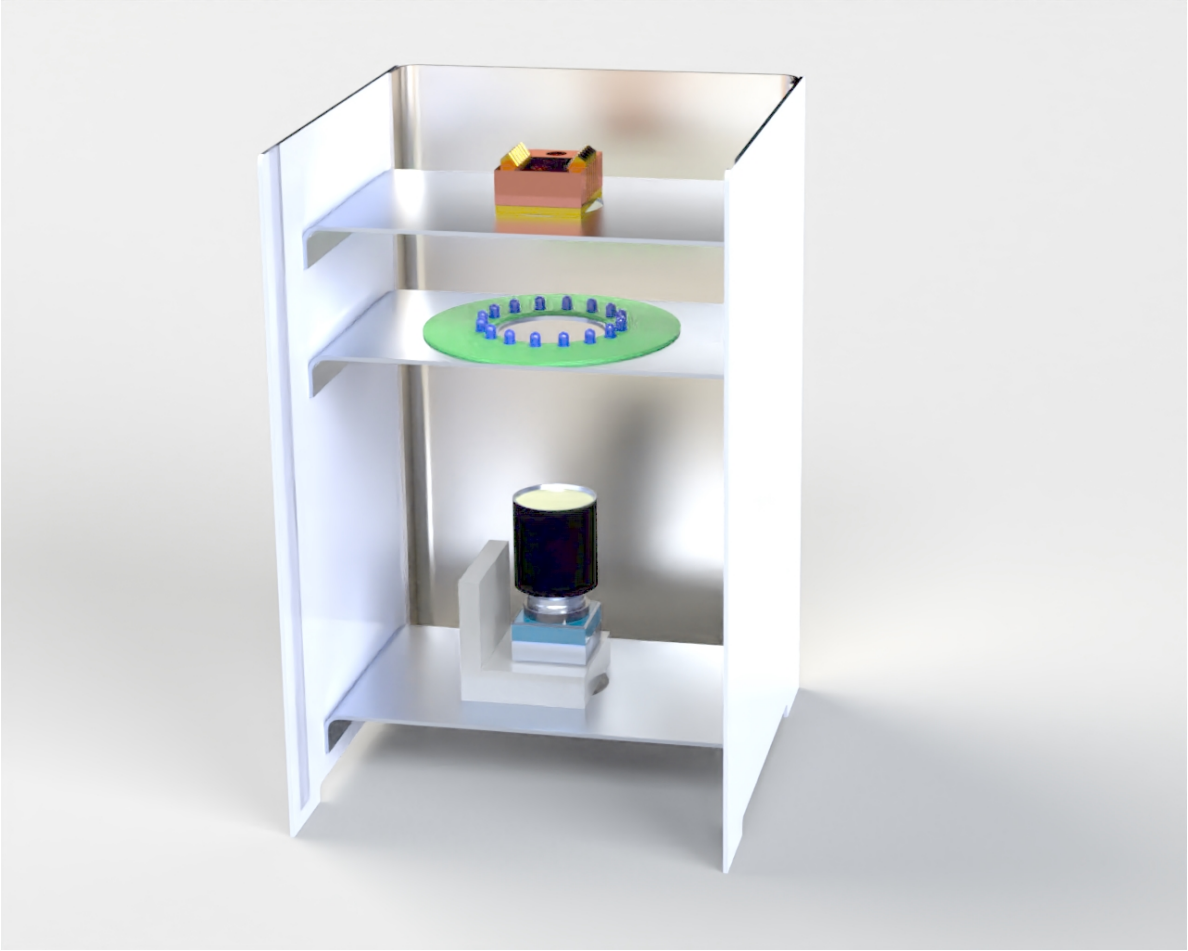

Figure 1: The prototype machine is a vertical optical bench with a special camera at the base and an LED lightened sample holder. The blood sample box at the top consists of six test chambers called targets where some observations occur along with a particular passage for the blood flow.

than others and requires alignment along the optical axis. Some shadow effects are also required to avoid posing further restrictions on the LED placement. However, that is not included in this work. Several academic studies using Extremal Optimisation were found in the search for methods to deal with problems like this, see the next section.

This report focuses on finding a valid starting point for a design problem using Extremal Optimisation (EO) techniques. The presented results do not represent an optimal configuration of the PCB, but more of a starting point from where established optimisation methods can be applied successfully. Primarily simulated annealing is believed to be the way forward in that. The method described here is the author's first interpretation of the Extremal Optimisation technique for solving combinatorial problems within the generative design. It has been found useful, and it is believed it can serve as a good tool for further studies on generative design or even automated

design, especially where the problem size and topology are part of the search.

This report is organised as follows:

- Sec. 3: General presentation of Extremal Optimisation.
- Sec. 4: Formulation of the test problem by transforming it from 3D to 2D.
- Sec. 5: Introduction of the concept of design flaw-improvement pair rules, **D2FI**.
- Sec. 6: Conceptual PCB rules.
- Sec. 7: Test results.
- Sec. 8: Discussion, observations, and software.
- Sec. 9: Conclusions.

3 Extremal Optimisation

By academic search services, one gets a clear view that Extremal Optimisation (EO) was an unknown business up until about 2000 when the early work of Boettcher et al. [1] started to get attention. Today one can see that most of the efforts using this search technique take place in China. Recent work indicates that EO could be a way forward in training certain types of neural networks for classification problems within the business of machine learning, [2]. Also, it seems that EO was first established as a heuristic approach to deal with some combinatorial problems within physics. Therefore, it is often referred to as a practical problem-solving method based on the statistical physics concept of self-organised criticality, SOC. The author has not studied the founding basis for SOC, but it seems fundamental to the mathematical description of natural phenomena like storms, earthquakes, granular piles, and even evolution itself.

A fundamental character of such a system is the built around barriers, obstacles that suddenly break down, resulting in avalanche progress of the system state. Furthermore, that is what attracts its principle for solving automated design problems as the progress of the solution often mimics such avalanche behaviour. The eye-opening article for the author was the first part of the article of Boettcher et al. [1] on re-partitioning network structures.

A wide variety of engineering problems have recently been addressed with techniques borrowed from EO. That includes load balancing in network configurations of both email communications systems and electrical grids [3] [4] [5] [6]. It also includes application within controller design [7]. It has also been studied for weight training in LSTM neural networks [8], which is interesting due to many parameters in such applications. The design of new proteins has been addressed using EO to overcome the extraordinary combinatorial problem size such biochemical engineering pose [9].

Most often are published academic work using EO in combination with more traditional optimisation methods. That is true for genetic algorithms [10] and particle swarm methods [11]. EO forms a way of addressing many-objective optimisation problems (MaOP) applications in combination with other methods [12]. Particularly in mechatronics, engineering problems are often found to include objectives from widely different fields [13].

4 Test problem formulation

The actual design problem for the prototype machine is still under a patenting process and therefore not revealed in detail here. The presented problem is an interpretation of the problem for the sole purpose of exemplifying Extremal Optimisation (EO) techniques for the combinatorial problem of placing electronic components onto a circuit board (PCB) with the restriction imposed by other mechanical, hydraulic, and optical restrictions in the machine prototype. The trace routing between the components is not included in the test problem but is a separate and later process. The electric current requirements of the PCB are less dominant, and therefore the width of the nest net traces are in the range of 12-20 mil and are believed to fit in between the component placed on the surface easily. The LEDs used have a significant overhang due to their optical lens size and then provided plenty of routing space beneath its footprint.

A record vector describes each LED and component according to:

$$(\mathbf{p}_{LED}, d_{LED}, R_{LED}, E_{LED}) \tag{1}$$

where $\mathbf{p}_{LED}$ denotes the (x, y)-coordinate of the component. d_{LED} is the component size, its diameter. R_{LED} is its operative range. It is further described bellow, but relate to the light emitting angular cone of the LEDs. E_{LED} is the emitting power in normalised form from each LED.

To simplify the coding of all components, they are treated equally, resulting in that the LED driver IC and the adjustment resistors (see figure 2) have no emitting power ($E_{LED} = 0$) and an range equal to a world size constant ($R_{world} = 100mm$). However, the emitting powers E_{LED} are not a part of the EO problem formulation but just included for further optimisation steps involving a global objective of equalising the received illumination at the six test chambers.

During the EO step sequence, the position $\mathbf{p}_{LED}$ of one component is updated based on some selection criteria, see section 5 below.

4.1 From 3D problem to a flat 2D description

To simplify the problem from a fully 3D problem based upon the placement of each LED in relation to the six test chambers, some approximations have been made.

Projecting LED cone: The LED has an emitting cone of light. These are deliberately selected to be of both wide-angle and narrow spotting types. Values go from 9° to 110°. Assuming the distance between the PCB surface and the test chambers is constant during the design, one can transform this property into an emitting range (R_{LED}) of each LED. This is the radius of the zone that is illuminated by the LED.

Neglecting optical axis extent: The test chambers have a significant extension in the optical axis direction. Chambers extent has been compressed into a set of six flat circles, as can be seen in the middle of the opening in 2. Further, the test chambers are considered only by their centers $\mathbf{p}_{c,n}$ where $n = [1..6]$.

LEDs aligned to optical axis: All LEDs are aligned with the optical axis, resulting in that the LEDs may very well be approximated with a circle and an omnidirectional equal emitting radius, as described in figure 2.

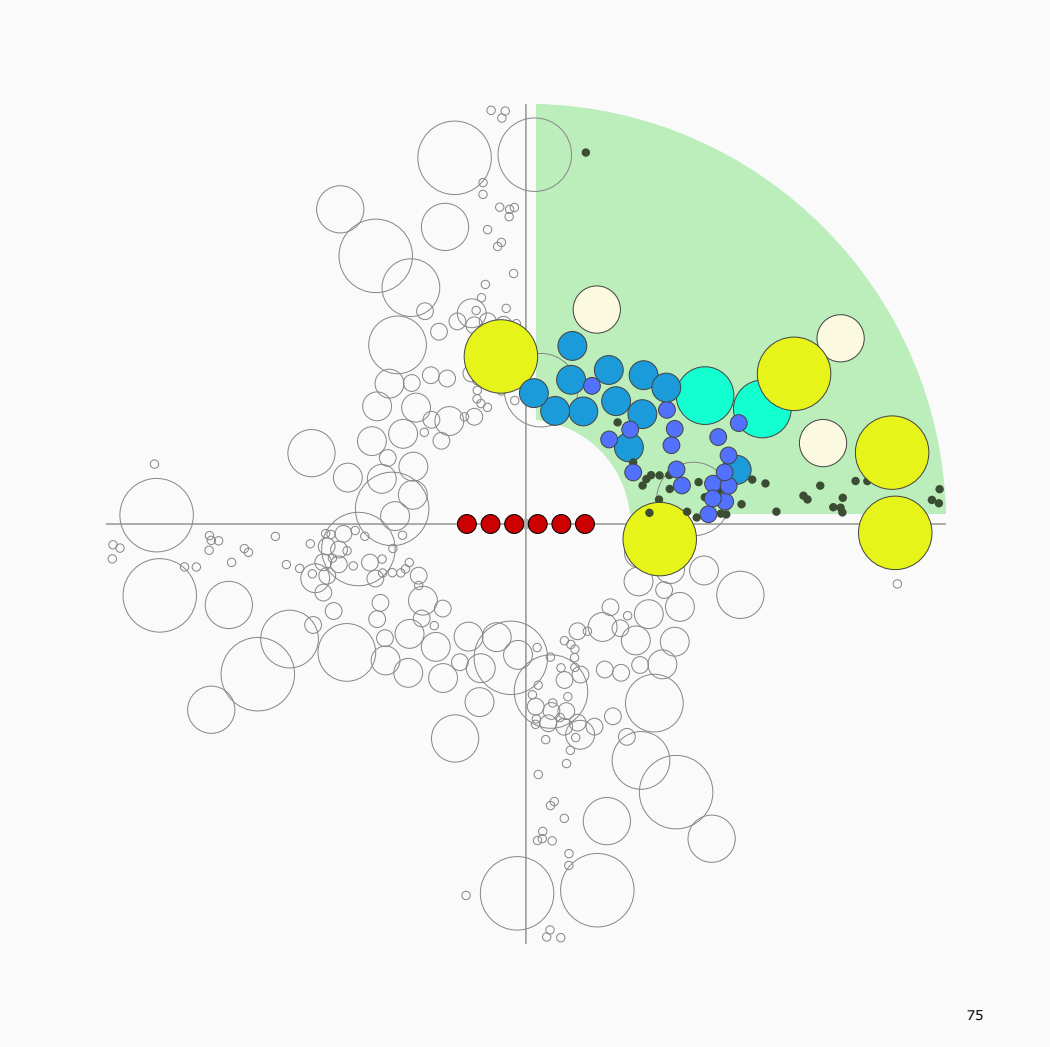

Figure 2: This is an intermediate result from a test run using EO. The problem is formulated onto a 2D quarter of a PCB surface with a hole in the middle for the optical axis. Symmetry is used for the PCB and is indicated by contours for quadrants 2-4. In the center are six test chambers placed, marked by red circles. The valid zone is the quarter-shaped segment in green, and the varying population of LEDs is seen in bluish color. The yellow circle marks LED driver IC-circuits that need to be in close vicinity of the LEDs connected and the small black dots represent current adjustment resistors that should go along each LED. In this view, a non-optimal layout has been reached.

5 Flaw-Improvment pair Rules, D2FI

In this work, the interpretation of Extremal Optimisation has been to improve the bad individual parts instead of the more classical approach in optimisation of promoting the good global solution. The algorithm used here can be described by two steps: identifying a flaw in the design and a corresponding improvement of that flaw. The pair of a design flaw and design improvement is called **D2FI** and forms a set of rules applied to the single design configuration repeatably.

The process may be described by two parts, a single topology and a multi-topology generative design process. The first focuses on reorganising a design with all its design variables defined at the beginning of the algorithm, and the second extends the procedure to make the topology of the design problem variable or scan multiple topologies sequentially.

5.1 Single Topology

Each flaw is identified sequentially by its type and a normalised score of flaw level. The number of possible types of flaws is fixed from the start, and they have a priority order between them, meaning some flaws are more significant than others and takes presence when it comes to the second stage of improvement. The flaws are ranked within each type class by the score of flaw level. In each step of the algorithm, one part of the design (one LED) is selected based upon its flaw score and is then improved based upon the priority of the type and flaw score level. This algorithm may get stuck in a local loop of **D2FI**-pair selections, but that is hindered by **not** selecting the flaw with the highest score each time. Instead, a probability density function is applied in the selection process favouring the most erroneous flaws for further improvement in most cases.

The overall algorithm is shown in algorithm 1 below. $C(\mathbf{x})$ represents the design space with the configuration $C()$ and its design variables $\mathbf{x}$. There are at most N design variables. The set of rules $\mathbf{R}_s$ denotes the possible identifiable design flaws f_s and their corresponding improvement action a_s. The design rules may be variable over the iteration of the algorithm, but the rules in the set $\mathbf{R}_s$ always remain normalised in the same fashion. There are at most $s \leq S$ design rules.

Algorithm 1 D2FI

Require: $C(\mathbf{x})$ $x = [x_1..x_N]$, $\mathbf{R}_s(f_s : a_s)$
Ensure: $N \geq 1, S \geq 1$
  $n \leftarrow 1$
  repeat
    $\texttt{sort}(C(\mathbf{x}), \mathbf{R}_s)$
    $(f_i : a_i) \leftarrow \texttt{select}(\mathbf{R}_s, p_s(n))$
    $\texttt{improve}(C(\mathbf{x}), a_i)$
    $n \leftarrow n + 1$
  until $f_s = 0, s \in S$

Of course, some other termination criteria may apply when trying to reach zero flaw score on all flaw types for all design variables. The iteration number n may be used or some threshold on some lower-priority flaw type.

The flaw score is in this algorithm assumed to be a scalar number between 0 and 1. It is seen in the following example that it is a bit of a challenge to encode the score in such a way. Moreover, the algorithm does not fail if the flaw score extends beyond this range, but it has proved important that all design parameters are bounded or have a limit based on an equation describing a parameter boundary.

The set of design rules S is evaluated sequentially, making one take precedence over the other in a falling scale of importance. It is believed that there will always be a way to establish an order among the possible design flaws, from devastating flaws to less catastrophic.

This hierarchy of rules and the introduction of a normalised value for the individual flaw levels form the main contribution in this study. This approach forms a difference from other rule-based search methods.

5.2 Multi-Topology

The multi-topology sequential search is a simple extension to the single topology evaluation described in the previous section. It adds or removes one specific set of design parameters if, and only if, the previous step of a single topology search results in a total design flaw of zero. Once the repetitive process of topology changes can not reach a zero design condition, the overall process stops, and the previous topology from the last being tested is taken as the result of the process. On the way, one gets a sequence of increasing or decreasing design complexity solutions.

6 The PCB design example

This section is an example of how to apply the algorithm 1 presented above. The following sub-sections are design flaws presented along with the related improvement implemented in this test case. Also, the normalisation of the flaw score is given. All parts are transformed into the 2D space limited by a maximum coordinate value of R_{world}.

6.1 Outside outer boundary

A circle represents the edge of the outer boundary. The design flaw score is represented by the red part of the component area compared to its total area (see figure 3). In position a is the component entirely outside the boundary used for identifying the design flaw. The flaw score may either be represented by the red circular area solely or combined with the projected light red area towards the boundary. In position b, is the component partly correctly placed and therefore the flaw score may be in the range [0..1]. In this case, a suitable flaw score is defined by the red part of the total circular area of the component. Finally, in position c the design flaw score is 0 since the component is entirely inside the valid zone.

$$f_i = \frac{4A_{outside,i}}{\pi d_{LED,i}^2} \tag{2}$$

Placement of a component making it cross the boundary in figure 3 will result in some geometric expression giving the area of the out of boundary part of the component circle (coloured red). The normalised score in eq. 2 then represents the flaw made by design configuring the component at position (x_c, y_c) on the x-y-plane. The score may be extended by the almost rectangular area seen in position a in figure 3. This is a minor deviation from the general idea of having all design flaw scores normalised to the range [0..1] but will be useful for quickly move far outplaced components into the valid zone.

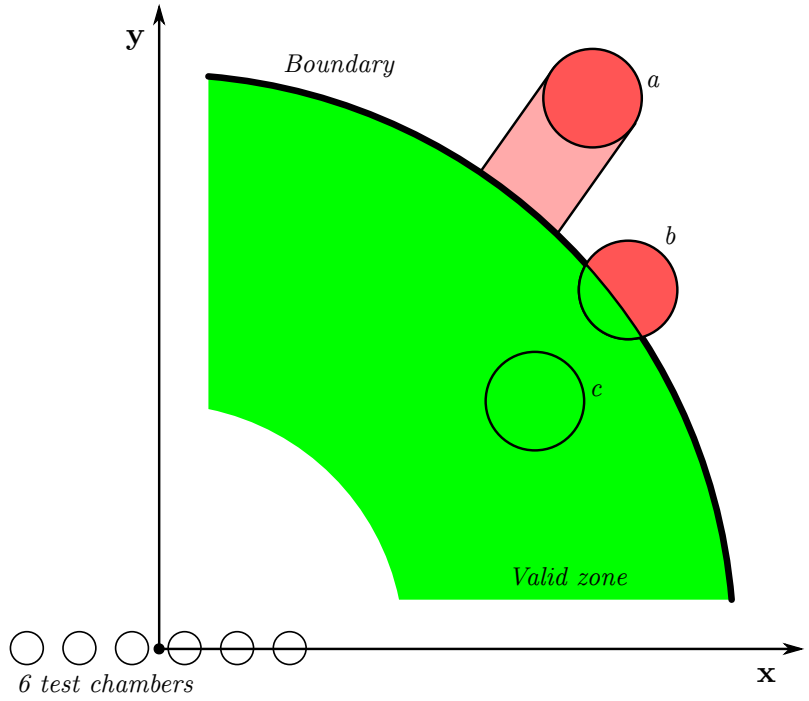

Figure 3: The quarter-shaped area forms the valid zone for component placement. Here one component is represented by its potential three positions a, b and c. Down to the left are the six test chambers indicated just for reference. That is where the blood flow passes through the machine. Symmetry is used, and the quarter-shaped zone is repeated by rotation to the other quadrants.

Once the flaw score has been selected and a component's position (x_c, y_c) has been selected for an update using this type of flaw, a position updating rule is applied.

$$(x_c, y_c) \leftarrow K_{f1} \times (x_c, y_c) + \mathbf{w}(n) \tag{3}$$

where $\mathbf{w}(n)$ represents a small random 2D vector to avoid cyclic behaviour of the algorithm further. Since the natural attractor for the improvement, in this case, is the PCB center, the value of K_{f1} is typically in the range of [0.80..0.99].

6.2 Inside inner boundary

In a comparable fashion as the outer boundary, a design flaw score may be established. In this case, the outer world maximum distance R_{world} does not need to be considered, and instead, the corresponding factor K_{f2} should be in the range of [1.01..1.20]. In this example, the largest component circle size is $14mm$ compared to the PCB center hole of $40mm$. Due to the difference in these two measures, there might be a problem with the applied randomness through a $\mathbf{w}(n)$ addition. An extended design flaw value related to the rectangular-shaped area in figure 3 position a has not been used or tested.

6.3 Outside left boundary

This flaw and improvement pair is slightly more straightforward to implement than the previous ones. Only the x_c coordinate of the position of the component needs to be considered. Some geometric mathematical expressions are needed to deal with components close to the corners of the quarter-shaped zone, but overall the expressions are simple. The improvement update function is relatively straightforward:

$$x_c \leftarrow d_{f3} + x_c \tag{4}$$

where d_{f3} represent a small scalar value in the range of [0.01..0.10].

6.4 Outside bottom boundary

This is similar to the boundary violation rules set above, except the objective coordinate is y_c instead, hence the improvement update function become:

$$y_c \leftarrow d_{f4} + y_c \tag{5}$$

where d_{f4} mimics the characteristics of d_{f3} above for symmetry reasons.

6.5 Overlapping avoidance

This rule differs from the first flaw and improvement pairs above since it involves the relation between one LED and the others on the board. Apart (LED or another component) is selected ((x_c, y_c)) based upon the amount of overlaid interferences from other components. It is then very straightforward; first, it searches for the closest neighbour in terms of center-to-center distance (x_n, y_n) that interfere with its boundary. Then both the nearest neighbour and the selected part are moved away by a small amount based on the direction given by the center-to-center vector. This neighbouring search can be costly and violates the idea of only considering design parts individually, but it is used here for simplicity and can be replaced by local search methods. The

total overlapping area is normalised towards the component's area, meaning that the flaw score may become greater than one but is still bounded by all the LEDs' total area. This mimics the normalising process from the rules presented above.

$$(x_c, y_c) \leftarrow k_{f5}\left((x_n, y_n) - (x_c, y_c)\right) \tag{6}$$

A scaling coefficient k_{f5} is used to affect the rate of convergence in this flaw-improvement pair. This set of rules will be the most used in the process, hence the importance of this coefficient's convergence rate.

6.6 Evenly light distribution for test chambers

Once the LEDs start to organise, the positions reflect the characteristics of the individual LED. Each LED has a parameter for its operative range R_{LED} as described above. In a similar way to the flaw-improvement pair above (eq. 6) the selected LED's position is adjusted towards the targets if it has a flaw that indicates that its R_{LED} range is less than the distance to any test chambers. Only when it reach 3 different chambers it is assumed not to have a flaw anymore concerning test chamber lightning. The improvement rule then becomes:

$$(x_c, y_c) \leftarrow k_{f6}\left((x_{st}, y_{st}) - (x_c, y_c)\right) \tag{7}$$

where the parameter st indicates one of the six targets. The target with the minor light emitted to it is selected by st. This varies with the position of the selected LED $((x_c, y_c))$ and the emitting power from all other LEDs. The enlightenment level for the targets is updated continually during the process. It is calculated by a radius-square relation for all six test chambers ($n = [1..6]$):

$$P_{target,n} = \sum_{k=1}^{N_{LEDs}} \frac{E_{LED,k}}{\left((x_t, y_t) - (x_c, y_c)\right)^2} \tag{8}$$

Here the normalised flaw only takes values in the set of $(0, 1, 2, 3)$ so no area-based normalising takes place.

7 Evaluation tests

This section describes a test run of the algorithm on the PCB design for a machine described in the introduction. There are several LEDs marked as coloured circles in the following diagrams. The largest circle describe the LED driver circuit and has no light emission but is instead a regular IC. Its occupancy area is still treated as a circle even if it has a rectangular shape. The valid PCB-surface is pictured in light green and forms the valid design space. The overall design space is a square with $100mm$ sides.

7.1 Single Topology Example

The example results come from a test run with a fixed number of LEDs, a single topology. It runs for 500000 state changes or steps. In each step, one LEDs is selected, most often the one with the worst total flaw score, but 0.5% of the times the second-worst is selected, and 0.005% of the steps is the third worst selected. This is to make the algorithm not get stuck in a bistable scenario.

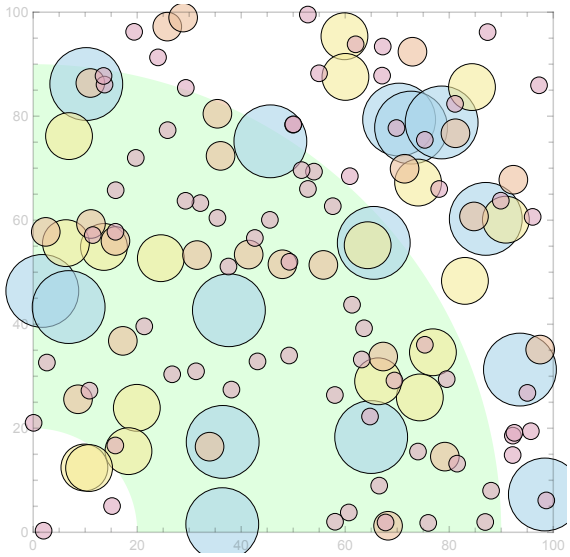

Figure 4: Starting topology from the beginning of the algorithm. The set of LEDs are randomly placed and selected from three categories. 72 pcs of 3mm LEDs (red small circles), 24 pcs of 5mm LEDs (orange larger circles), 18 pcs of 8mm LEDs (yellow large circles). To balance this, a number of LEDs 16 pcs of driving circuits, is added as well (blue largest circles). In the background, one can see the valid PCB surface in light green. This board layout forms step 1 in the process of 500000 steps.

Observations: The exemplified test run takes 50 seconds on average to complete using an Intel i7 CPU at 4.6GHz. The memory footprint is tiny.

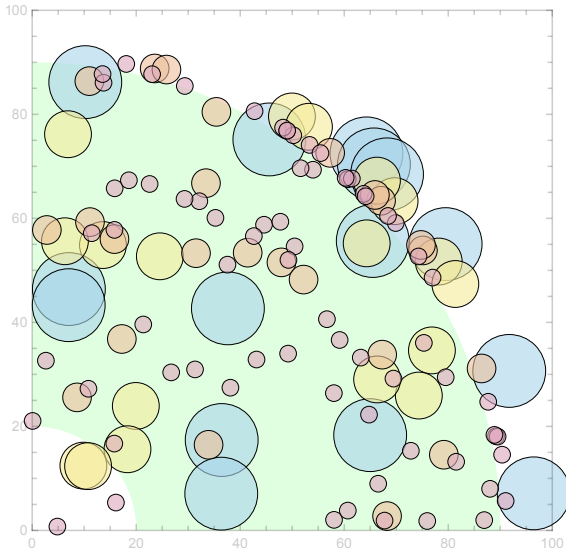

Figure 5: This is step 924, and the hierarchy of flaws can be observed in that the outliers are drawn into the valid PCB-area first. The LEDs placed inside the green area have only minor position adjustments, while those outside the PCB have moved onto or close to the PCB border. Some minor changes on the left and bottom border can also be observed.

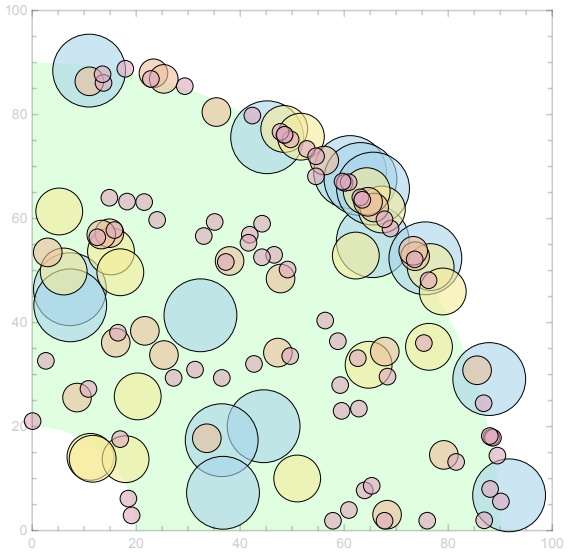

Figure 6: This is step 1852 and the set of flaw-improvement pair rules $\mathbf{R}_s$ have been applied equally many times, bringing the process further. Some effects of the overlay flaw can be observed since some larger circles have bounced off the other border of the PCB. At the same time, the other borders have attracted outliers more.

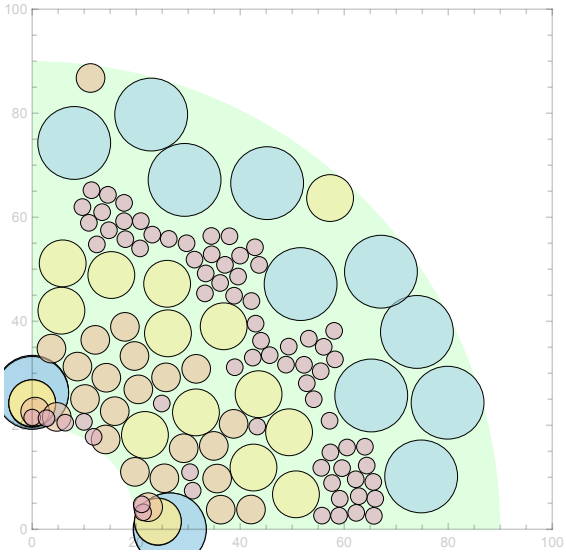

Figure 7: This is step 66983, and some ordering among the LEDs can be seen. This is the last flaw-improvement rule coming into more application forwarding light emission onto the targets situated near origo in the figure.

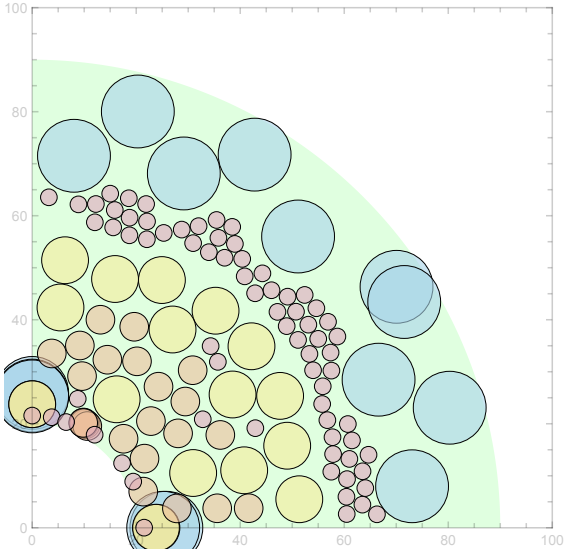

Figure 8: This is step 196264, and further order can be observed. Still, some overlay flaws are noticed especially close the corners of the inner circle of the PCB area.

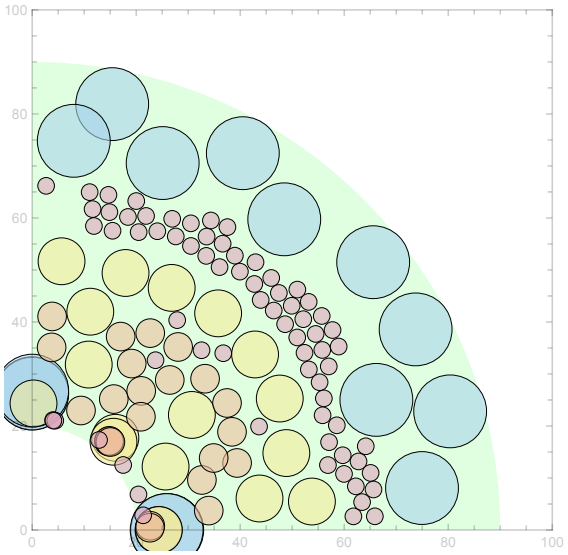

Figure 9: This is the final step 500000, and the process terminates. The overlapping issues remain for the circles in the close vicinity of the driver circuits (blue large circles). Still, the current state of the layout could serve as a good starting point for a optimisation using some global scheme like genetic algorithms or simulated annealing.

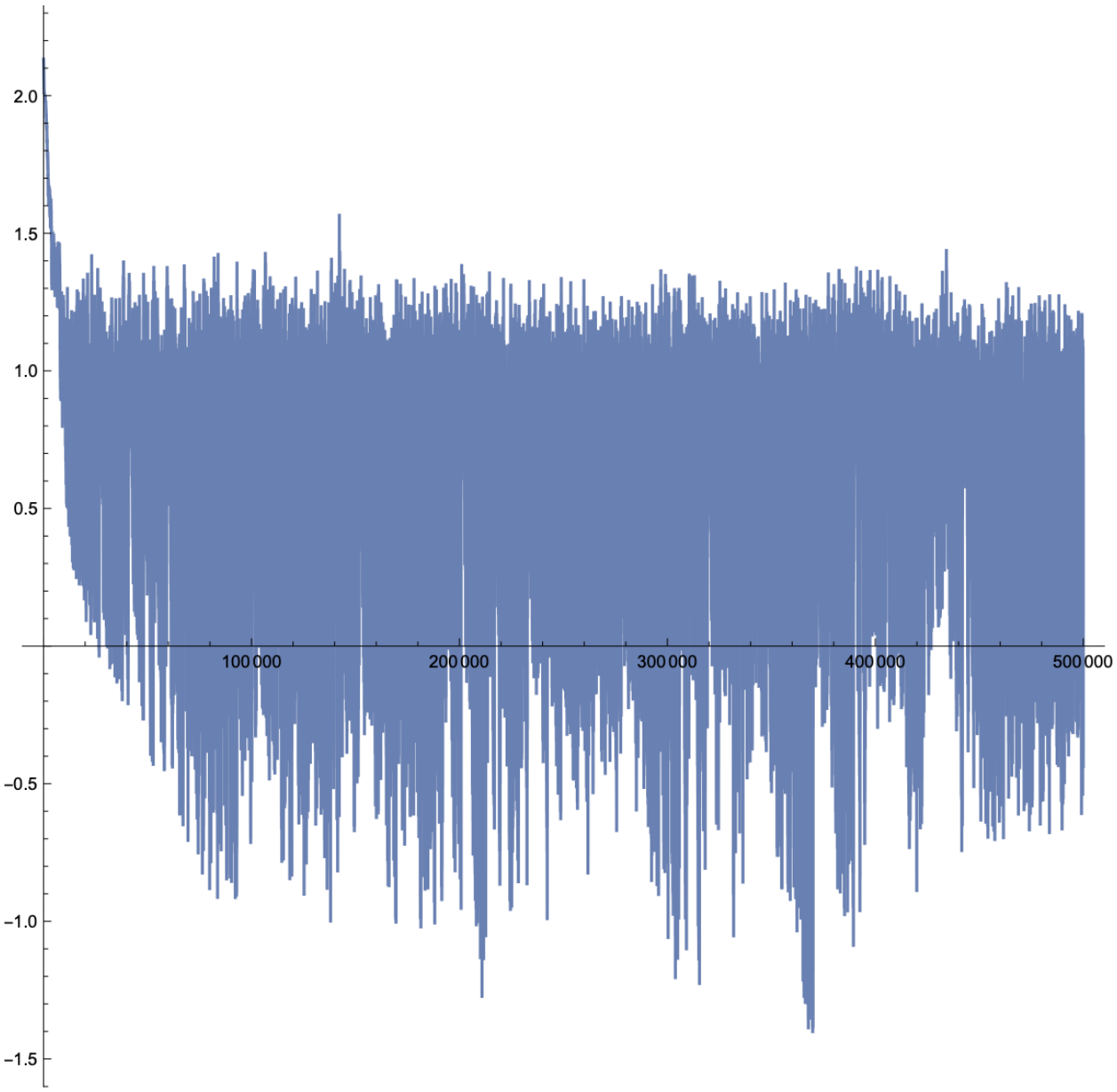

Figure 10: This is the total flaw score of the test run, where the score value is presented as a $\log_{10}$ value. It is not to be interpreted as a global design error value, revealing some 4 orders of magnitude reduction in the design flaws but is just presented as an indication that the algorithm works as a search method using flaw-improvement rule pairs. Most noticeable is that several deep pockets of very low total flaw scores are passed during the test run. Some thousand tests have been carried out, all converging to a result similar result to this diagram.

8 Discussion

This short and preliminary study has lead to many observations. It is worth mentioning that the study is intended as a search method for finding robust start conditions for further optimisation of the machine's LED layout and **not** as an optimisation method on its own. The idea is also to be able to start with a rather broad parameter search space. Similar to other academic works presented related to EO techniques.

The overall procedure is intended to support a flexible and quick method to reconfigure the machine's performance during the prototype development process. The lighting conditions of the test chambers or targets are the LED PCB's objective and need to be varied during different prototype stages. An alternative procedure to the one presented here could have been used applying modern CAD software in combination with optimisation tools. Several commercial tools exist for that.

It is noticeable that the algorithm 1 has a similar look as a classical genetic algorithm with the `sort()-select()-recombine()` structure. This may be extended in future versions to include branches in the search space, making the LEDs positions not only just being shifted by a small step on the PCB but moved away to a new area entirely.

Such movement needs to take all other LEDs into account, which violates the proposed method's character to only evaluate individual components attributes when looking for the highest design flaw score in each step. A small violation of that has already been made since the search for the most overlapping design flaw evaluates all LEDs relation to each other. Nevertheless, there exist ways to reduce this $O(n^2)$ search by modern algorithms. The unique approach in this algorithm to only evaluate each component's design flaw individually encourages parallel execution. Still, in this early stage, the program has been implemented as a single-threaded program unit using C++14 template structures, and no parallelism has been applied.

The total flaw score seen in figure 10 above indicates that the organising of the LEDs' position works. A considerable number (1000+) of similar test runs have been carried out, and none have got stuck in a limiting loop. This is due to the introduction of small randomness when selecting the individual with the highest flaw score, as described in section 7.1 above. The results presented in figure 10 are not to be interpreted as a minimised design error even if it looks like the flaw goes down with an increased number of steps. Instead, the correct interpretation of it is that the search of the design space works. One can see from the logarithmic values that the total flaw score goes down from around 130 to 0.04, a drop of ca 3300 times or nearly four orders of magnitude.

One may find many steps in the algorithm, and the presented example uses 500000 steps, overwhelming. However, consider that a standard global optimisation method usually requires 1000 to 10000 steps for a 150-variable optimisation problem. That also results in some state changes among variables in the range of millions.

Finally, a more subjective remark can be made on this work. To make this flaw-improvement pair ruleset work, one needs to have an algorithmic approach to describing the problems at hand. This may differ from standard optimisation methods where one formulate mathematical equations that are to be minimised. Perhaps one can find it more challenging to establish this set of rules than defining a global mathematical

expression, and also, the computational performance of the method depends highly on the implementation of the rules. The selection of high-performance computer languages like C++ has become very fruitful, just by chance. However, languages like Haskell maybe even more suitable for future studies.

8.1 Next step

The presented method is going to be used further in the project of creating this blood analysis machine. It has been combined with simulated annealing with promising results already, but some more analysis on the combined results is still needed.

8.2 Software release

The software created in this project has been made available for public use with the restriction of no commercial use and attribution according to the Creative Commons license model. The use of it is described in the attached LICENCE file. Please notice that it is highly specialised and requires some good C++ insight.
GitHub link: `https://github.com/magse/pcb_layout_demo_eo`.

8.3 Referencing

Attributing this work is primarily done by referencing the indicated scientific work as indicated on the `https://github.com/magse/pcb_layout_demo_eo` site.

8.4 Acknowledgment

The author presented this report at the SICFP21 online conference in June 2021. The author happens to be the conference's information system organiser, controlling the peer-review process in detail. For that reason, no peer-review process has been carried out on this contribution.

9 Conclusion

This report presents preliminary results from the mechatronic design process of a blood analysis machine that combines engineering from highly diverse fields like chemistry, electronics, fluid power, optics, and computers. This report focuses on the preprocessing stages for optimising the layout of LEDs onto a PCB surface. The following observations were made:

- Combining design flaws and their related improvement/updating actions into a pair forms a good structure for algorithmic generative design and possible for automatic design.
- The process of updating a mechatronic design stepwise is done by selecting one action from a hierarchical set of rule pairs where the most severe design flaw score takes presence over the rest, on a falling scale of severeness.
- The algorithm's performance indicates that the avalanche barrier crossing characteristics found in self-organised criticality (SOC) systems are inherited into the presented **D2FI** algorithm. The search for such characteristics was the initial driver for this study.
- A set of six rules organised hierarchically has proved to work when each rule flaw is normalised into a value of [0 .. 1].
- Bounded deviations of the normalising flaw score are allowed and do not affect the overall performance, just the sequencing of the improvements taken.
- The procedure has been exemplified upon a combined mechatronic design problem of optics and target lightning conditions where the original geometric 3D problem has been reduced to a flat 2D domain.
- The algorithm have been executed many times without any observation of being stucked in a loop condition, mainly due to some randomness in the flaw selection step.
- Further investigation on the combination with a following global optimisation procedure is still needed.
- The current state of demonstration software using the proposed approach has been made available through GitHub.

10 References

References

[1] Stefan Boettcher and Allon G. Percus. Extremal optimization: Methods derived from co-evolution. 1999.

[2] F. Zhao, G. Zeng, and K. Lu. Enlstm-wpeo: Short-term traffic flow prediction by ensemble lstm, nnct weight integration, and population extremal optimization. *IEEE Transactions on Vehicular Technology, Vehicular Technology, IEEE Transactions on, IEEE Trans. Veh. Technol*, 69(1):101 – 113, 2020.

[3] Ivanoe De Falco, Eryk Laskowski, Richard Olejnik, Umberto Scafuri, Ernesto Tarantino, and Marek Tudruj. Dynamic load balancing based on multi-objective extremal optimization. *2020 19th International Symposium on Parallel and Distributed Computing (ISPDC), Parallel and Distributed Computing (ISPDC), 2020 19th International Symposium on*, pages 134 – 141, 2020.

[4] Noemi Gasko, Mihai Alexandru Suciu, Tamas Kepes, and Rodica Ioana Lung. Shapley value and extremal optimization for the network influence maximization problem. *2019 21st International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2019 21st International Symposium on*, pages 182 – 189, 2019.

[5] Yi-Yuan Huang, Min-Rong Chen, Liu-Qing Yang, Kang-Di Lu, and Guo-Qiang Zeng. An electric load forecasting model based on bp neural network and improved bat algorithm hybridized with extremal optimization. *2019 Chinese Automation Congress (CAC), Chinese Automation Congress (CAC), 2019*, pages 4673 – 4678, 2019.

[6] Jing-Liao Sun, Xiao-Qing Xie, Ru Xiong, Guo-Qiang Zeng, Huan Wang, and Yu-Xing Dai. Binary-coded extremal optimization based fractional-order frequency control of an islanded microgrid. *2017 36th Chinese Control Conference (CCC), Control Conference (CCC), 2017 36th Chinese*, pages 10679 – 10685, 2017.

[7] Guo-Qiang Zeng, Xiao-Qing Xie, Min-Rong Chen, and Jian Weng. Adaptive population extremal optimization-based pid neural network for multivariable nonlinear control systems. *Swarm and Evolutionary Computation*, 44:320 – 334, 2019.

[8] Min-Rong Chen, Bi-Peng Chen, Guo-Qiang Zeng, Kang-Di Lu, and Ping Chu. An adaptive fractional-order bp neural network based on extremal optimization for handwritten digits recognition. *Neurocomputing*, 391:260 – 272, 2020.

[9] Keiichi Tamura, Hajime Kitakami, Tatsuhiro Sakai, and Yoshifumi Takahashi. A new distributed modified extremal optimization for optimizing protein structure alignment. *2015 IEEE 8th International Workshop on Computational Intelligence and Applications (IWCIA), Computational Intelligence and Applications (IWCIA), 2015 IEEE 8th International Workshop on*, pages 109 – 114, 2015.

[10] F. Pistolesi, B. Lazzerini, M.D. Mura, and G. Dini. Emoga: A hybrid genetic algorithm with extremal optimization core for multiobjective disassembly line balanc-

ing. *IEEE Transactions on Industrial Informatics, Industrial Informatics, IEEE Transactions on, IEEE Trans. Ind. Inf*, 14(3):1089 – 1098, 2018.

[11] Fleford Redoloza and Liangping Li. A comparison of extremal optimization, differential evolution and particle swarm optimization methods for well placement design in groundwater management. *Mathematical Geosciences*, 53(4):711 – 735, 2021.

[12] Min-Rong Chen, Guo-Qiang Zeng, and Kang-Di Lu. Constrained multi-objective population extremal optimization based economic-emission dispatch incorporating renewable energy resources. *Renewable Energy*, 143:277 – 294, 2019.

[13] Leandro dos S. Coelho, Viviana C. Mariani, Rafael B. Grebogi, Emerson H. de Vasconcelos Segundo, Mauricio V. Ferreira da Luz, Jean V. Leite, and Roberto Z. Freire. Diversity-guided generalized extremal optimization for transformer design problem. *2017 IEEE Symposium Series on Computational Intelligence (SSCI), Computational Intelligence (SSCI), 2017 IEEE Symposium Series on*, pages 1 – 6, 2017.