Layout Synthesis for Symmetrical Facades

Andres Felipe Barco Santa, Élise Vareilles, Michel Aldanondo, Paul Gaborit

Layout Synthesis for Symmetrical Facades:
Constraint-Based Support for Architects Decision-Making

A. F. Barco, E. Vareilles, M. Aldanondo, and P. Gaborit

Université de Toulouse, Mines d'Albi
Route de Teillet Campus Jarlard, 81013 Albi Cedex 09, France
Corresponding author: abarcosa@mines-albi.fr

Abstract. Facade-layout synthesis problem deals with the allocation of an *undetermined number* of rectangular parameterizable panels over a rectangular facade surface. Requirements state that panels must not overlap, must be placed in specific supporting areas and must cover existing windows and doors over the facade. Due to the constrained constitution of the problem, constraint satisfaction and constraint programming come naturally as solving techniques. However, as most constraint programming environments use as arguments a well-defined set of variables, speculation about the number of panels to be allocated becomes a critical issue for its automation. On this regard, we present a two-phase solution: First determine the structure of the layout-plan and second, pass to a constraint solver a fully declarative model using such structure. We show that our solutions are consistent over symmetrical facades, and thus, can be used for early stages of architectural design. Our goal is to assist architects with a constraint-based support system.

Keywords: Layout Synthesis, Constraint satisfaction problems, Architects decision-making, Decision support systems

1 Introduction

The problem. Currently buildings energetic consumption represents more than a third part of the total energy consumption in developed countries [6, 8, 17]. One strategy for reducing such energy consumption lies on buildings thermal retrofit achieved either by an internal or an external insulation [12]. Among several options [12], an external insulation may be based on covering the entire building with an envelope made out of rectangular wood-made panels [9, 22]. However, some difficulties exist when targeting such retrofit in industrial proportions, e.g. in a country. These difficulties include slow conception using by hand configuration, human scheduling and craft assembly. Thus, it is essential to assist this massive retrofit of buildings with decision support systems [13].

Now, although finding a correct allocation of entities over a given surface can be efficiently made by a human, it is not the same for finding all solutions or finding an optimal solution. In fact, finding an optimal solution among the set of possible solutions would require a significant amount of time if the wrong technique is used due to the combinatorial properties of the problem. Simply stated,

the problem is NP. The definition is as follows. Given as input a rectangular facade surface and size limitations for rectangular panels, create a layout-plan solution to cover the entire facade. A layout-plan solution is an assignment of size (width and height) and position to each panel in such a way that all facade and panel related requirements are respected. Size limitation for panels result from manufacturing conditions and are translated into lower and upper bounds for panels width and height. The problem includes three characteristics never considered simultaneously: It deals with the allocation of an *unfixed number* of rectangular parameterizable panels that must not overlap, frames (existing windows and doors) must be *overlapped* by one and only one panel, and facades have specific areas providing certain *load-bearing capabilities* that allow to attach panels. As far as we know, only a previous work by the authors that uses a greedy approach [2] have been proposed to address this problem. That solution, however, generates only one valid solution for a given facade specification and does not involve optimality criteria.

Related work and contribution. Literature relevant to solve the problem can be found on layout synthesis [14] and rectangle packing [11]. The work in [23], where an apartment space is divided into a matrix where a finite set of rooms with predefined size are allocated, inspires our solution. However, as it does not deal with unfixed number of entities; it cannot be used to solve our problem. The same drawback is found in [4], where authors present a constraint-based framework to tackle layout synthesis problems in a 2D reference plane. Although geometrical entities have undetermined size their number is known, and thus, is not appropriated for our problem. Two-dimensional packing literature also presents studies relevant to solve our problem [11]. Nevertheless, only few studies tackle the problem of undermined number of variables (rectangles). In fact, such scenario have been addressed using several approaches such as new consistency methods [21], exploration techniques [3] or combination of possibilistic [19] or weighted [7] constraint satisfaction to find solutions [10]. However, these kinds of methodologies are not implemented in most constraint programming environments and thus expertise for using existing technologies is required. Regarding the geometrical constraint *Geost* [5], we consider it to complex because we only deal with rectangular shapes. Also, given that our industrial application may evolve, we rely on our constraint-based implementation.

The aim of this paper is to propose a first strategy for a constraint-based layout synthesis over symmetrical facades. Then, we present a declarative model stating all constraints and the objective function, a process to generate compliant layout plans, and some experimental cases using a support system. The support system is able to generate different solutions for a given facade specification. The paper is divided as follows. In Section 2, the facade retrofit elements and our assumptions are introduced. In Section 3, the constraint satisfaction problem (CSP) definition of the problem is presented. In Section 4, the solving process and the construction procedure are introduced. Experimental cases, using a Java prototype [1] that uses Choco [18] as underlying solver, are provided in Section 5. Some conclusions are discussed in Section 6.

2 Building Retrofit

Facades. A facade (presented in Figure 1) is represented by a two-dimensional coordinate plane, with origin of coordinates (0,0) at the bottom-left corner of the facade, and contains rectangular zones defining:

- Perimeter of facade with its size (height and width in meters).
- Frames (windows and doors) which play an important role as they are meant to be overlapped by one and only one panel. Frames are defined by: Origin point (x,y) with respect to origin of facade, width and height (in meters).
- Supporting areas. As the layout problem must deal with a perpendicular space plan, gravity must be considered. It turns out that some areas over the facade have load bearing capabilities that allow us to attach panels. Supporting areas have well-defined: Origin point (x,y) with respect to origin of facade, width and height (in meters).

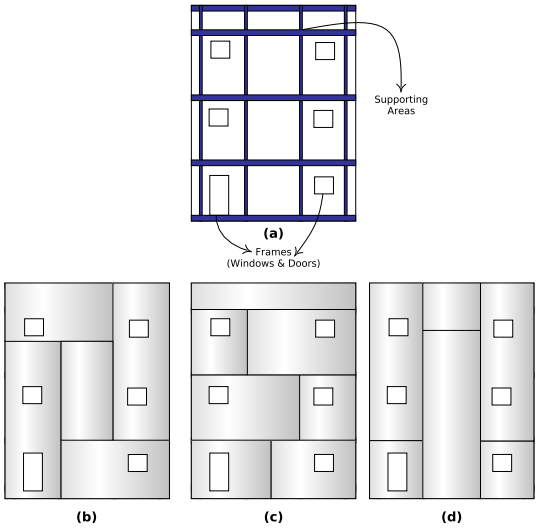


Fig. 1. Facade to renovate and valid insulating envelopes.

Rectangular panels. Panels (see Figure 2) are rectangular, of varying sizes and may include different equipment (solar modules, window-holes, shutters, etc.). These panels are designed one at a time in the process of layout synthesis and manufactured in the factory prior to shipment and installation on the building site. These panels have a well-defined:

- Size (height and width). The size is constrained by a given lower and upper bound consequence of working-site or manufacturing limitations.

– Thickness and insulation. Thermal performance of a given panel depends on several properties: Size, thickness and insulation type. Consider that the smaller the thickness of the panel the better quality should be the insulation in order to reach performance objectives.
– New frames (such as new doors and new windows). Given internal structure of rectangular panels, new frames must respect a parameterizable minimum distance (Δ) with respect to panel's borders.
– Cost depending mainly on size and attached equipment (in Euros).
– Thermal performance (in watts per meter square-kelvins, $w.m^{-2}.k^{-1}$) depending on size, chosen thickness and insulation type.

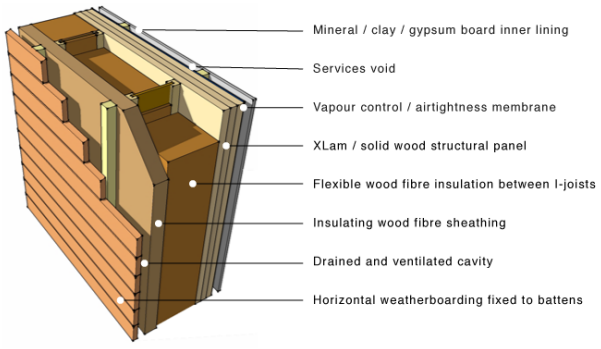


Fig. 2. Rectangular parameterizable panel.

Key limitations. As mentioned in the introduction, there are three key issues reflected from the industrial scenario, been the unfixed number of panels the more problematic one. Considering the internal structure of panels, partially overlapping a frame is forbidden and, frames' border and panels' border must be separated by a minimum distance denoted by Δ. Additionally, in oder to attach panels, the corners of each panel must match a supporting area.

Assumptions. Our solution is conceived to tackle symmetrical facades, i.e., those facades in which supporting areas are uniformly distributed over the surface and in which frames are symmetrically arranged on the surface as shown in literal (a) of Figure 1. Two assumptions have been made for the present work. First, all supporting areas are strong enough in such a way that the problem only deals with the placement of panel's corners over supporting areas. In other words, there are unlimited load-bearing capabilities in supporting areas and no capabilities in the remaining of the facade surface. Second, in order to use a simple yet intuitive objective function, we assume panel's thickness to be constant and we consider only one type of panel's insulation.

3 Constraint model

A CSP is described in terms of a set of variables $\mathcal{V}$, a collection of potential values $\mathcal{D}$ for each variable and a set of relations $\mathcal{C}$ over those variables, referred to as constraints [15, 16]. A constraint is a relation representing partial information over the variables of the problem. A CSP solution is an assignment of values for each variable in such a way that all constraints in $\mathcal{C}$ are satisfied.

Thus, to give the constraint-based solution a clear focus and to formalize the facade-layout synthesis problem as a CSP, we have to cope with:

- Decision variables describing the layout solution,
- constraints describing the relations over panels and facade.

In addition, we present our objective function appropriated for the retrofit industrialization.

3.1 Constraint variables

We introduce the notation used in the model. Let F denote the set of frames and S the set of supporting areas. Let $o_{e.d}$ and $l_{e.d}$ denote the origin and size, respectively, of a given entity e in the dimension d, with $d \in \{1, 2\}$ (1 for x-axis and 2 for y-axis). For instance, $o_{fr.1}$ denotes the origin in the horizontal axis and $l_{fr.1}$ denotes the width of frame fr. Additionally, lb_d and ub_d denote the size lower bound and size upper bound, respectively, in dimension d for all panels.

Intuitively, each panel is described by its origin point with respect to the facade origin and its size. For convenience, let us assume that $\mathcal{P}$ is the set of panels composing the layout-plan solution. Then, each $p \in \mathcal{P}$ is defined by $\langle o, l \rangle$ where:

- $o_{p.d} \in [0, l_{fac.d}]$ is the origin of panel p in dimension d.
- $l_{p.d} \in [lb_d, ub_d]$ is the size of panel p in dimension d.

3.2 Constraints

The following six constraints express the main relations among panels, and between panels and facade that must respect a layout solution.

(a) *Manufacturing and transportation limitations constrain panel's size with a give lower bound* `lb` *and upper bound* `ub` *in one or both dimensions.*

$$\forall p, 0 \leq p < n, d \in \{1, 2\} \quad lb_d \leq l_{p.d} \leq ub_d$$

(b) *For two given panels p and q there is at least one dimension where their projections do not overlap.*

$$\forall_p \in P, \forall_q \in P, p \neq q, \exists d \in \{1, 2\} \mid o_{p.d} \geq o_{q.d} + l_{q.d} \vee o_{q.d} \geq o_{p.d} + l_{p.d}$$

(c) *A given panel p must either be at the facade edge or ensure that enough space is left to fix another panel.*

$$\forall_p \in P, d \in \{1,2\},\ o_{p.d} + l_{p.d} = l_{fac.d} \vee o_{p.d} + l_{p.d} \leq l_{fac.d} - lb_d$$

(d) *Each frame over the facade must be completely overlapped by one and only one panel. Additionally, frames' borders and panels' borders must be separated by a minimum distance denoted by Δ.*

$$\forall_f \in F, \exists p \in P, d \in \{1,2\} \mid o_{p.d} + \Delta \leq o_{f.d} \wedge o_{f.d} + l_{f.d} \leq o_{p.d} + l_{p.d} + \Delta$$

(e) *The entire facade surface must be covered with panels.*

$$\sum_{i \in P} \prod_{d \in \{1,2\}} l_{i.d} = \prod_{d \in \{1,2\}} l_{fac.d}$$

(f) *Panels' corners must be matched with supporting areas in order to be properly attached onto the facade.*

$$\forall_p \in P, d \in \{1,2\}, \exists s \in S \mid o_{s.d} \leq o_{p.d} \wedge o_{p.d} + l_{p.d} \leq o_{s.d} + l_{s.d}$$

3.3 Objective function

In the industrial scenario, the ranking is made with respect to cost and thermal performance of the layout plan. To simplify computations, and assuming a constant thickness and one insulation type, the cost and thermal performance are only computed with respect to panels' size. Thus, the cost of a panel p is computed with $c_p = (l_{p.1} \times l_{p.2}) * (\alpha - l_{p.1} - l_{p.2})$ where α is a factor provided by the panels' manufacturer which depends on panels material and internal structure. Note that the term $(\alpha - l_{p.1} - l_{p.2})$ decreases with the size of the panel and thus manufacturing large panels is less costly, globally, than manufacturing small ones.

Additionally, due to the thermal characteristics of the retrofit, the less panels' junctions the better: It is at panels' junctions that the bigger thermal transfer exists. In consequence, given the upper bound for panels' size, facades should wear panels as large as possible while respecting the architectural constraints, supporting areas, manufacturing and working site conditions. An optimization function which entails optimal cost and optimal thermal performance is the maximization of panels' areas

$$maximize\Big(\sum_{p \in P} (l_{p.1} \times l_{p.2})\Big) \tag{1}$$

4 The solving process

Dealing with the problem of having an unfixed number of panels (e.g. variables) is prerequisite for solving the problem. Thus, the solution have been divided in two phases: First find the structure of the layout plan by finding an appropriated number of panels to allocated (Section 4.1) and, second, launch constraint solving over the selected number of panels using the aforementioned declarative model (Section 4.2).

4.1 Providing structure to the plan

Given the lower and upper bounds for panel's size and the facade size, we may compute the minimum and maximum number of panels that can be fixed in the surface. For instance, the theoretical minimum number of panels that can be fixed in a given dimension d is nb_h=$l_{fac.d}/\text{ub}_d$ if ($l_{fac.d}$ mod $\text{ub}_d == 0$) and nb_h=($l_{fac.d}/\text{ub}_d$)+1 otherwise. The minimum number of panels over the facade surface is the product of minimum number of panels that can be fixed on each dimension.

4.2 Construction procedure

The main difficulty for solving the problem using Choco is that the number of panels is unknown. It is a difficulty because, as well as many other constraint programming environments [20], Choco uses a well-defined set of variables and constraints to execute search. Our proposed procedure, presented in what follows, creates a matrix of panels to cover the facade. To do so, it uses the minimum number of panels that can be fixed over the facades surface to define the layout-plan structure and then iteratively adds and constrains each panel with respect to the constraint knowledge and with respect to the already placed panels. If the number of panels is not enough the model is executed again using one additional panel.

Step 1:- Compute the minimum number of panels that can be placed both horizontally and vertically (nb_h, nb_v).
Step 2:- Create an array of $\text{nb}_h \times \text{nb}_v$ constraint variables representing areas.
Step 3:- Do a double loop with nb_h and nb_v to construct a *matrix* of panels. For each inner loop creates the corresponding decision variables: Origin ($o_{p.1}$,$o_{p.2}$) and its size ($l_{p.1}$,$l_{p.2}$). For the first panel in the layout, ($o_{p.1}$,$o_{p.2}$) is assigned to (0,0).
  - Step 3.1:- Initiate decision variables ($o_{p.1}$, $o_{p.2}$, $l_{p.1}$, $l_{p.2}$) in respective domains (constraint (a)).
  - Step 3.2:- Constrain points ($o_{p.d}$,$o_{p.d} + l_{p.d}$) to be inside supporting areas (constraint (f)).
  - Step 3.3:- Link panel area to $\prod_{d\in\{1,2\}}(o_{p.d} + l_{p.d})$
  - Step 3.4:- Post non-interference constraint (constraint (c)).
  - Step 3.5:- Post constraint for frames overlapping (constraint (d)).
  - Step 3.6:- If there is a previous panel q in x-axis, ensure non-overlapping (constraint (b)) by setting origin point $o_{p.1}$ of this panel to be the end point of previous panel $o_{q.1} + l_{q.1}$.
  - Step 3.7:- If there is a previous panel q in y-axis, ensure non-overlapping (constraint (b)) by setting origin point $o_{p.2}$ of this panel to be the end point of previous panel $o_{q.2} + l_{q.2}$.
  - Step 3.8:- Post surface covering constraint (constraint (e)).
  - Step 3.9:- Add decision variables to solution.
Step 4:- Select a search strategy and apply search to the previous decision variables and maximizing decision variables areas.

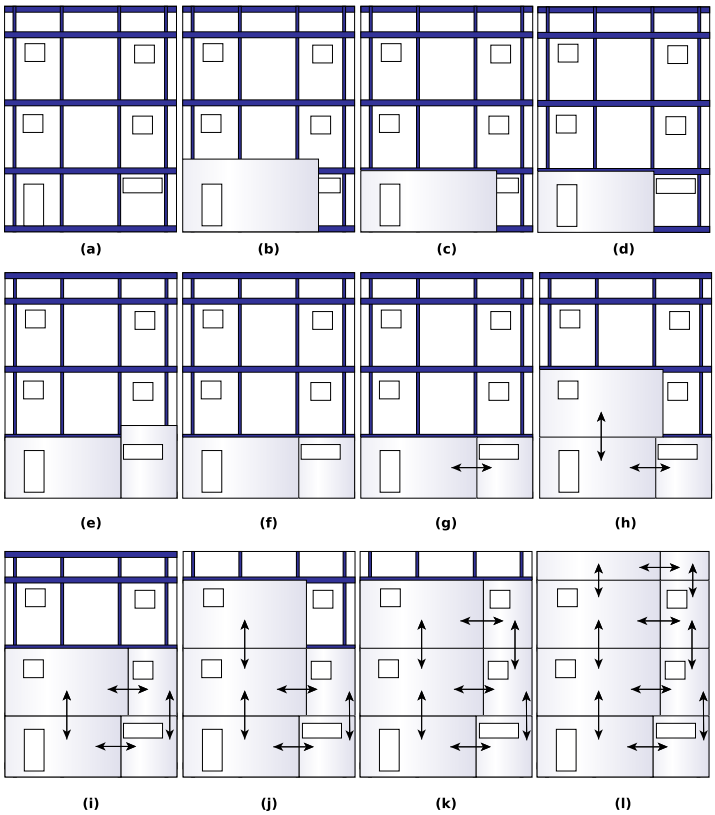


Fig. 3. Layout construction using procedure.

An illustration of the procedure's behavior is shown in Figure 3.

As commented in the procedure, the origin point ($o_{p.1}$,$o_{p.2}$) of the first panel in the layout is deterministically assigned to (0,0). Two reasons support our choice. First, there is always a supporting area in (0,0), otherwise the total facade surface cannot be covered. And second, it avoids the use of symmetry breaking constraints. In fact, as each panel is indistinguishable from the others, the procedure places each panel in the first available point next to the previous placed panels.

States from Fig.3(b) to Fig.3(f) present the results for applying supporting areas (constraint (f)) and frames overlapping (constraint (d)) constraints. From Fig.3(g) to Fig.3(k), the Figure illustrates the relation between each panel and the previous placed panels that avoids overlapping. Finally, State in Fig.3(l) shows the complete layout plan. Note that panel's width and height are set by the heuristic `minDom_UB` trying the maximum of their domain.

5 Experimental cases

The proposed procedure have been tested over several simulated facades and real facades. Figure 4 present statistics for fictitious facades on execution time for first solution, number of solutions found over a time window of 10 seconds, number of backtracks and explored nodes in the same time window. As we are dealing with symmetrical facades, frames and supporting areas are uniformly distributed over the facade surface. Also, facades have enough supporting areas in such a way that is possible to attach an envelope, i.e., the problem has a solution. Configuration for panels are; $lb_1 = 0.9$ meters as width lower bound and $ub_1 = 13.5$ as width upper bound; $lb_2 = 0.9$ meters as height lower bound and $ub_2 = 3.5$ meters as height upper bound. Additionally, as we are using integer domains, values are parsed to integers by means of simple arithmetic operations.

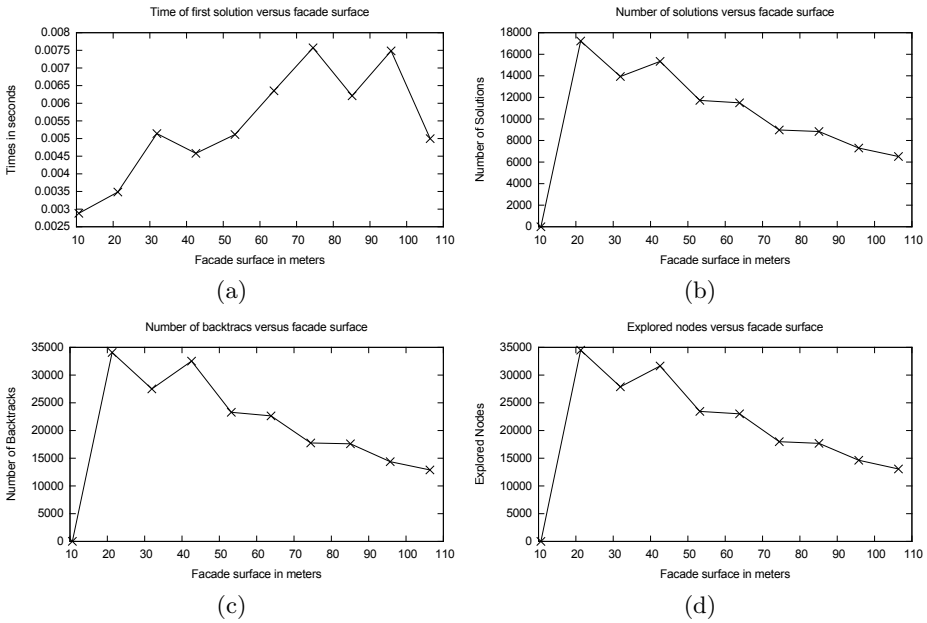


Fig. 4. Execution statistics.

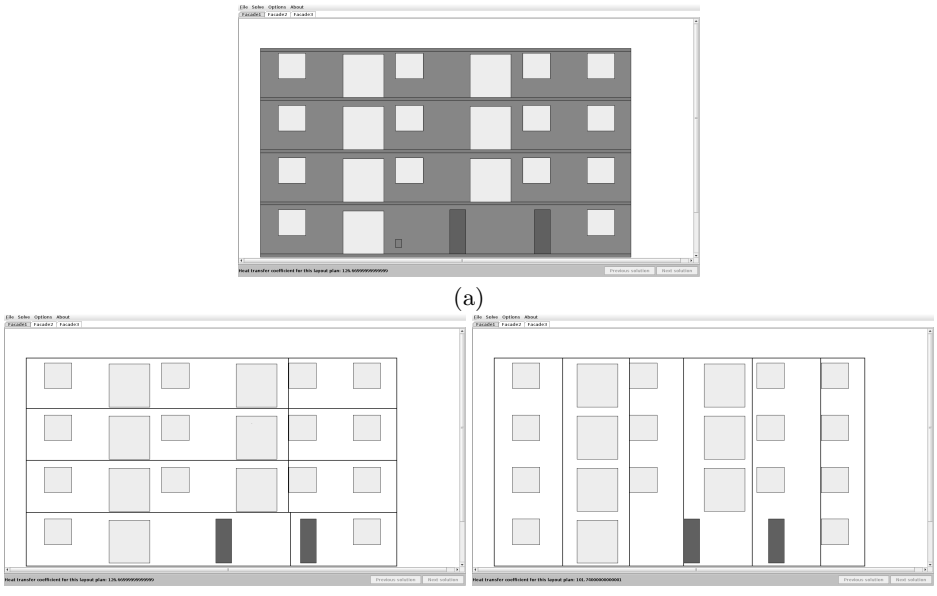
From the graphics is deduced the following. As expected the number of solutions found in 10 seconds decreases with the facade surface size. This is due to increasing of the number of relations when increasing the number of panels. Additionally, the execution time depends on the relation between facade size and panels' upper bounds. For when maximizing the area of each panel, the solver would fail less, and thus backtrack less, when panels can be fixed using their size upper bounds. Recall that the same upper bound have been used for all tests.

Nonetheless, regardless the relation between facade's size and panels' size, the time for finding a solution is competitive for the industrial scenario where no real-time interaction is needed and the user (e.g. an architect) has enough time to run and select the appropriated panels-made envelope.

Now we present some solution examples of particular instances. The following examples illustrate real-life facades part of the working site *La Pince* in the commune Saint Paul-lès-Dax in the department of Landes, France. In each example, panels lower bounds are $lb_d = 0.9$ meters for both dimensions $d \in \{1, 2\}$. Size of facade in literal (a) Figure 5 is $l_{fac.1} = 18.95$ and $l_{fac.2} = 10.64$ meters, it has 23 frames (21 windows and 2 doors) and only horizontal supporting areas.

Example 1. Layout plan in literal (b) is generated with panels' upper bounds of $ub_1 = 13.5$ meters and $ub_2 = 3.5$ and its structure is a matrix with dimensions 4×2. The solution is found in 0.056 seconds, 36 nodes were explored and 27 backtracks executed.

Example 2. Literal (c) shows a layout plan generated with $ub_1 = 13.5$ meters and $ub_2 = 3.5$ meters and a matrix structure of 1×6. The first solution is found in 1.182 seconds, 86191 nodes where explored and 86184 backtracks executed.



(a)

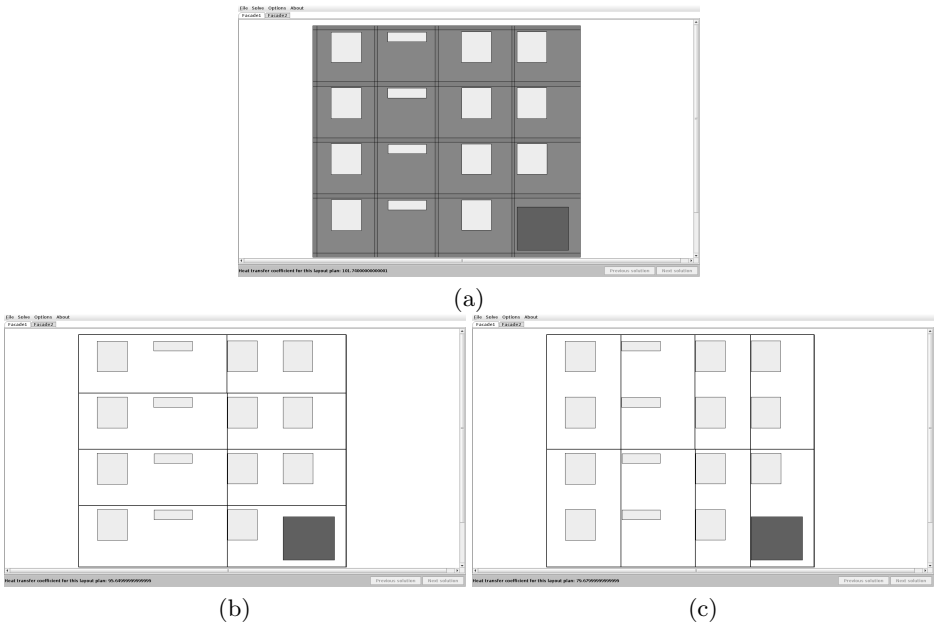
(b) (c)

Fig. 5. First example for real facade.

Size of facade in literal (a) Figure 6 is $l_{fac.1} = 12.598$ and $l_{fac.2} = 10.907$ meters, and it has 16 frames (15 windows and 1 door).

Example 3. The layout plan in literal (b) is generated using panels' upper bounds of $ub_1 = 7$ meters and $ub_2 = 3.5$ with an structure of 4×2. The first solution is found in 0.023, 10 nodes where explored and 1 backtrack executed.

Example 4. Literal (c) shows a layout plan generated using $ub_2 = 7$ meters and $ub_1 = 3.5$ meters and thus its matrix structure is 2×4 panels. The first solution is found in 1.071, 97019 nodes where explored and 97010 backtracks executed.



(a)

(b) (c)

Fig. 6. Second example for real facade.

As the results present, finding a compliant solution using vertical panels (i.e., height > width) takes more time, explores more nodes and backtracks more than horizontal panels because the possible attaching points (i.e., supporting areas) are more numerous. In fact, the presented facades have horizontal supporting area every 2.5 meters, approximately. Thus, using horizontal oriented panels for those facades permits only one possible supporting area to attach them whereas using vertical panels ones leads to several possible attaching points.

All the presented solutions where found by Choco solver by invoking the `findOptimalSolution` method. By contrast, the system finds 2544 solutions invoking the `findAllOptimalSolutions` method over facade in literal (a) Figure 5. Many of these solutions have not a significant difference as they may differ only in one centimeter for a given panel. However, in order to assist architects decision-making it is important to present a reduced number of solutions and let

them, given aesthetics aspects or other criteria, the choice on what solution to implement.

6 Conclusions

The reduction of energy consumption in buildings is recognized as an international key issue for the coming years. As the construction of new energy efficient buildings is not enough to face the demand, the retrofit of existing ones is a necessity. In addition, conception and implementation of such retrofit must be supported by intelligent systems if efficiency and optimal solutions are desired.

This work is part of a project that investigates the possibility of automated building retrofit based on rectangular panels and assisted by a support system. Within the project, a greedy solution have been proposed [2] but it is able to generate only one solution without applying any notion of optimality. We have presented our first approximation for solving the layout synthesis of symmetrical facades using a declarative approach that includes:

1. A description of the (constraint) knowledge supporting this layout problem as a CSP,
2. presented a solution procedure that incorporates all of the identified constraints and,
3. illustrated its behavior on an example from the pilot project site.

The constraint-based solution presented in this paper follows a two-phase approach to generate facade-layout plans: Find the minimum number of panels to use and then pass a declarative model to generate layout plans using a constraint solver. Layout solutions are always a matrix whose dimensions depend on the width and height of the facade and panels. Evidently, a matrix solution is a mayor drawback given that they will generate valid solutions only in symmetrical facades. So, despite the benefits provided by constraint technology with respect to to the number of solutions thrown, efforts for targeting asymmetrical facades appear to be an important follow-up of this work. Then, a declarative solution for non-symmetrical facades is currently under development as well as a methodology for presenting to architects a pertinent (aesthetic constraints) subset of all possible layout-plan solutions.

Constraint programming is well-suited for addressing this industrial problem because on the one hand, its declarative view allows a clear knowledge representation, and on the other hand, the building of a prototype using an open constraint programming environment is much easier, thanks to all the pre-defined constraints, search and provided abstractions. We have shown that our solutions are consistent over symmetrical facades, and thus, can be used for early stages of architectural design.

References

1. Barco, A.F., Vareilles, E., Aldanondo, M., Gaborit, P.: Calpinator: A configuration tool for building facades. In: 16th International Configuration Workshop. pp. 47–54. CEUR Workshop Proceedings (2014)
2. Barco, A.F., Vareilles, E., Aldanondo, M., Gaborit, P.: A recursive algorithm for building renovation in smart cities. In: Andreasen, T., Christiansen, H., Cubero, J.C., Raś, Z. (eds.) Foundations of Intelligent Systems, Lecture Notes in Computer Science, vol. 8502, pp. 144–153. Springer International Publishing (2014)
3. Barták, R.: Dynamic global constraints in backtracking based environments. Annals of Operations Research 118(1-4), 101–119 (2003)
4. Baykan, C., Fox, M.: Spatial synthesis by disjunctive constraint satisfaction. Artificial Intelligence for Engineering, Design, Analysis and Manufacturing 11(04), 245–262 (1997)
5. Beldiceanu, N., Carlsson, M., Poder, E., Sadek, R., Truchet, C.: A generic geometrical constraint kernel in space and time for handling polymorphic k-dimensional objects. In: Bessière, C. (ed.) Principles and Practice of Constraint Programming – CP 2007, Lecture Notes in Computer Science, vol. 4741, pp. 180–194. Springer Berlin Heidelberg (2007)
6. Center, T.E.C.: Energy Conservation Handbook. The Energy Conservation Center, Japan (2011)
7. Cooper, M., De Givry, S., Sanchez, M., Schiex, T., Zytnicki, M.: Virtual arc consistency for weighted csp. In: Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 1. pp. 253–258. AAAI'08, AAAI Press (2008)
8. Council, U.G.B.: New Construction Reference Guide (2013)
9. Falcon, M., Fontanili, F.: Process modelling of industrialized thermal renovation of apartment buildings. eWork and eBusiness in Architecture, Engineering and Construction pp. 363–368 (2010)
10. Faltings, B., Macho-Gonzalez, S.: Open constraint programming. Artificial Intelligence 161(1-2), 181–208 (2005)
11. Imahori, S., Yagiura, M., Nagamochi, H.: Practical algorithms for two-dimensional packing. Chapter 36 of Handbook of Approximation Algorithms and Metaheuristics (Chapman & Hall/Crc Computer & Information Science Series) (2007)
12. Jelle, B.: Traditional, state-of-the-art and future thermal building insulation materials and solutions - properties, requirements and possibilities. Energy and Buildings 43(10), 2549 – 2563 (2011)
13. Juan, Y., Gao, P., Wang, J.: A hybrid decision support system for sustainable office building renovation and energy performance improvement. Energy and Buildings 42(3), 290 – 297 (2010)
14. Liggett, R.: Automated facilities layout: past, present and future. Automation in Construction 9(2), 197–215 (2000)
15. Montanari, U.: Networks of constraints: Fundamental properties and applications to picture processing. Information Sciences 7(0), 95–132 (1974)
16. Olarte, C., Rueda, C., Valencia, F.: Models and emerging trends of concurrent constraint programming. Constraints 18(4), 535–578 (2013)
17. Pérez-Lombard, L., Ortiz, J., Pout, C.: A review on buildings energy consumption information. Energy and Buildings 40(3), 394 – 398 (2008)
18. Prud'homme, C., Fages, J.: An introduction to Choco 3.0, An open source java constraint programming library. In: CP Solvers: Modeling, Applications, Integration, and Standardization. International workshop. Uppsala Sweden (2013)

19. Schiex, T.: Possibilistic constraint satisfaction problems or "How to handle soft constraints?". In: Proceedings of the Eighth International Conference on Uncertainty in Artificial Intelligence. pp. 268–275. UAI'92, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1992)
20. Schulte, C., Carlsson, M.: Finite Domain Constraint Programming Systems. Chapter 14 of Handbook of Constraint Programming (Foundations of Artificial Intelligence). Elsevier Science Inc., New York, NY, USA (2006)
21. Van Hoeve, W., Régin, J.: Open constraints in a closed world. In: Beck, J., Smith, B. (eds.) Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, Lecture Notes in Computer Science, vol. 3990, pp. 244–257. Springer Berlin Heidelberg (2006)
22. Vareilles, E., Barco Santa, A., Falcon, M., Aldanondo, M., Gaborit, P.: Configuration of high performance apartment buildings renovation: A constraint based approach. In: Industrial Engineering and Engineering Management (IEEM), 2013 IEEE International Conference on. pp. 684–688 (Dec 2013)
23. Zawidzki, M., Tateyama, K., Nishikawa, I.: The constraints satisfaction problem approach in the design of an architectural functional layout. Engineering Optimization 43(9), 943–966 (2011)