SPATIAL SIMILARITY METRICS

Graph theoretic distance measurement and floor plan abstraction

THOMAS GRASL AND ATHANASSIOS ECONOMOU
Georgia Institute of Technology, USA

Abstract. Comparing space allocation programs computationally is a resource intensive task. This paper introduces a method which reduces the complexity of floor plan graphs to facilitate the problem. In a first step the nodes are labeled according to a laid out classification scheme. This in its own right reduces the complexity of the comparison by introducing more diversity among the nodes. Subsequently the graph can be coarsened in several steps based on the category information. This does not only further reduce the necessary computation, but can also offer a visualization of and additional insight into the essential structure of the space allocation program at hand.

1. Introduction

Analyzing architectural composition and concluding on the expected performance has always called for systematic and reproducible methods. The translation of the architectural plan into a less ambiguous representation in form of a graph was the first step in creating a whole toolbox of methods following this ambition (March and Steadman 1974). Not only do graphs represent architectural layouts in a lean and computable manner, one can also take advantage of proven methods from a broad area of research in graph and network theory.

The computational and algorithmic complexity of many methods in these areas has often restricted their use for applications in the field of architecture. New algorithmic and electronic developments are however continuously extending the possibilities in this area of research.

The work presented in this paper modifies the graph theoretic methods of multilevel graph manipulation (Karypis and Kumar 1995) and error-tolerant graph matching (Bunke and Allermann 1983) to analyze buildings with complex space allocation programs. The corpus of buildings, that is used

A Dong, A Vande Moere & JS Gero (eds), CAADFutures'07, 251-263.

here to test the methodology and the application, consists of the federal Courthouses built since 1990. These buildings are complex structures, conceived as sorting machines, which are organized around explicit functional requirements.

The wider research project on the description and evaluation of new federal Courthouses has been supported by the GSA and the US Courts. The specific work presented here utilizes a graph representation of the courthouse plans to arrive at a distance measure for space allocation programs. Subsequently the intention is to expand the work into the foundations of a systematic theory on the typology of federal Courthouses.

Section 2 introduces the collection of methods used to arrive at the similarity metric. The problem of deriving the graph from the plan is discussed and an approach to facilitate computation through node labeling is introduced. The background to both the graph collapsing and the distance metric are explained before finally stepping through the implemented algorithm.

Section 3 briefly discusses the real world application for analyzing courthouses. This work is currently in progress and advancements are made continuously. Some findings from the first iterations are however discussed and more is soon to follow.

Finally, the infamous future work section describes some work currently being implemented and also an extended set of possibilities left open for the time being.

2. Method

The following section discusses how to arrive at the distance measure between two space allocation programs. Once the graph is created a method to collapse adjacent and similar nodes is introduced to enable comparison of the graph on various levels of coarseness, thereby enabling distance measures which are not dominated by minor differences in spatial composition, but which capture essential typological similarity. All graph computations are illustrated on the simple pictorial example shown in figure 1, a floor plan taken from the work on Palladio by Stiny and Mitchell (1978).

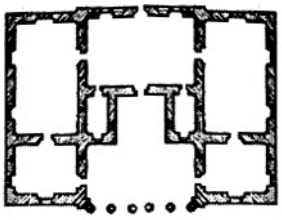

Figure 1. A floor plan in the Palladian style (Stiny and Mitchell).

2.1. DERIVING THE GRAPH

In order to represent floor plans as graphs the method described by Steadman (1976) was adopted. Generally it can be said that areas are mapped to nodes and relations between these areas are mapped to edges.

Any part of the floor plan may be considered to be an area represented in the graph (Figure 2); typically areas are defined by clearly denoted boundaries in the representation of the floor plan. It must however be realized that, the recognition of areas is not always a trivial task; if clear boundaries are not present it is often a matter of interpretation. Where does the entrance hall end and the corridor begin? A set of rules must be defined for each research project in order to resolve such issues consistently. Several methods to resolve this dilemma have been proposed, the most frequently adopted solution being the convex partition (Berg et al. 2000; Hillier et al. 1984).

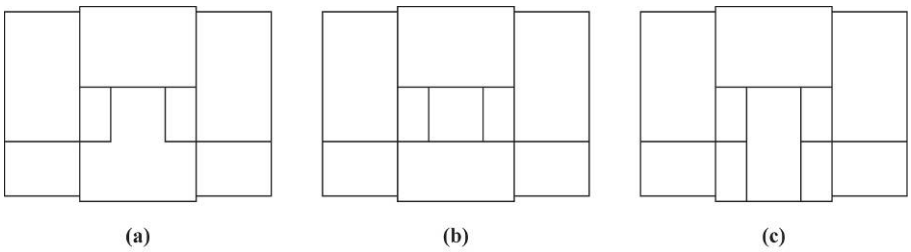

Figure 2. Alternative diagrammatic representations of the floor plan of Figure 1. a) 8 areas b) 9 areas; c) 10 areas.

Any relation between areas may be considered as a relation represented in the graph; adjacency (Figure 3-c) is the most frequently represented relationship. Steadman describes adjacency as sharing some length of wall in common.

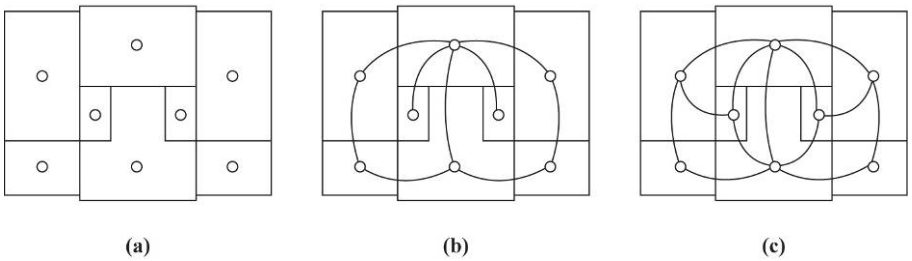

Figure 3. Alternative graph representations of the diagram of figure 2(a). a) Null graph; b) Accessibility graph; c) Adjacency graph. All graphs depict relations of areas within the plan and not with its carrier space.

Thereby, whether the complete set of adjacencies or only a subset thereof is used is a matter of interpretation and will most likely be dictated by the nature of the problem being investigated. One may be interested only in adjacencies which additionally allow for circulation (Figure 3-b), or on the extreme, in no relations at all (Figure 3-a).

The increasing use of building information models, and hence appropriate space information, will lead to future projects which allow for automated graph generation. This of course does not eliminate the necessity to follow conventions; it merely shifts the responsibility to the creator of the building information model.

2.2. AREA CATEGORIZATION

Each node is assigned a category label which is compiled from a combination of access level, area function and unique identification. The root element to which all nodes are assigned is the generic area category. Subsequently nodes are divided according to their affiliation to the various subcategories (Figure 4). The first level represents the access zones, which can be seen as top level separator of the various user flows. Airport terminals may distinguish between public, passenger, staff and secure zones, whereas hospitals may chose to differ between staff, patient and public areas. Other building typologies may be categorized in similar ways.

The second and third levels of category are based on functional distinctions and should be configured in such a way as to allow grouping of similar areas. It will later become evident, that the sooner a differentiation between two adjacent areas occurs, the less likely it is that these areas will ever unite and the greater the impact will be on the similarity measure.

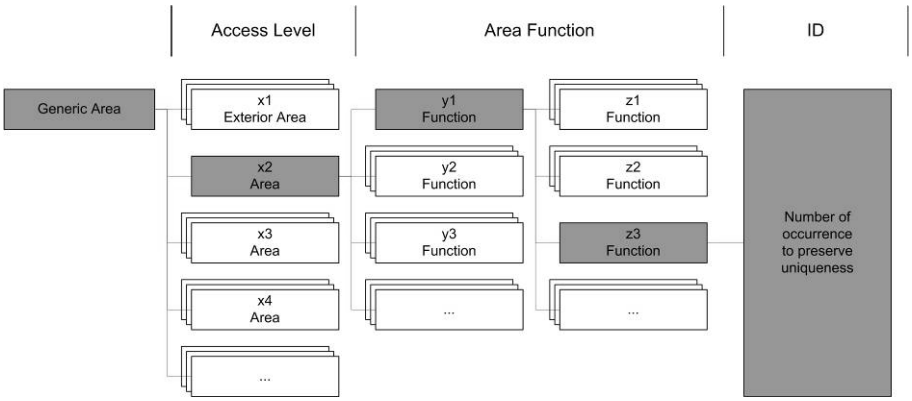

Figure 4. General structure of a category tree. The numbers of columns and rows of the data structure can be increased as desired.

Finally, in order to systematically distinguish between two areas even if they belong to the same category, a unique identifier is added as last element. Whether a global or local identifier is used is not of importance.

The example described above uses a four level category tree, it is however possible to extend the depth of the tree arbitrarily to better represent the typology at hand. Especially the description of the functional category may well require more than two levels.

Several standards of area categorization, such as ANSI-BOMA, E1836-01 and ISO 9836 exist, should either of these fulfill the requirements of the typology at hand it is advisable to implement these.

2.3. GRAPH COLLAPSING

Simplifying graphs before executing expensive calculations is a technique finding increasing use in the field of computer science. The most prominent example being multilevel graph partitioning (Karypis and Kumar 1995), a graph partitioning approach which iteratively coarsens a graph to a computationally feasible size before performing the partitioning operation. The intermediate result is then brought back to the original graph, back-tracking the coarsening steps and refining the result in the process. In such a way good results are achieved even for the un-coarsened, original graph, while computation is kept at a manageable level.

While collapsing a graph necessarily information is reduced, hence the method of collapsing depends entirely on the feature of the graph which should be sustained. In most cases the subtracted information can be stored in a separate data-structure in order to be reintegrated in a later step. Shortest path algorithms often prune cul-de-sac branches and leafs to simplify the search, should the source or target node be amongst the pruned nodes the relevant information is added from a look-up table.

The approach taken here is to collapse an edge e into a hyper-node w, if its incident nodes u and v belong to the same category (Figure 5-a). This may be done on several levels of coarseness, by varying the depth of required equality (Figure 5-b). The first step may be to require identical categories on three levels, then only the first two levels could be taken into consideration and finally it would suffice if the nodes belong to the same access level group.

Moderately complex space allocation programs may comprise 200-300 individual areas. Not only is it computationally expensive to perform comparisons on such graphs, it also includes the danger that the resulting value is dominated by topological noise and fails to capture the essential aspects of the buildings. Differently placed support areas such as closets and storage areas may distort the similarity of the overall typologies. It is of

course a matter of interest at which level one engages with an analysis, one must however be aware that there are computational limits.

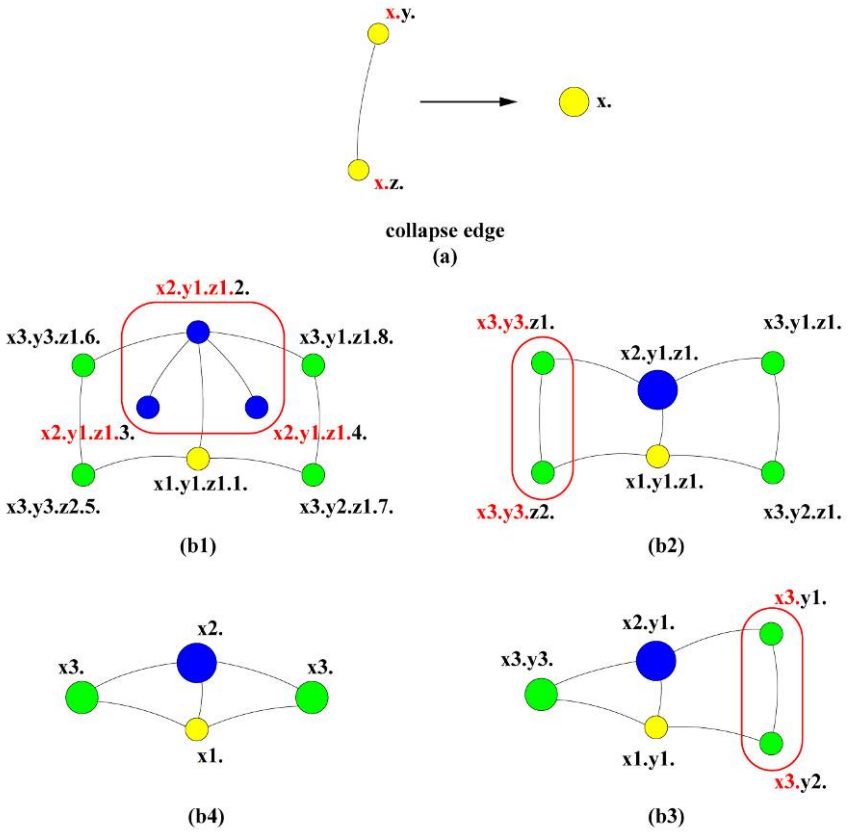

Figure 5. The collapse edge operation (a): The two nodes u→ x.y and v→ x.z collapse to a node w→x if u and v belong in the same category x. The new node w is represented with an area equal to the sum of the areas of the subsumed nodes. The application to a derivation: (b1) the original graph, (b2) graph collapsed to level 3, (b3) graph collapsed to level 2, (b4) graph collapsed to level 1.

2.4. DISTANCE METRIC

The problem of finding a distance metric to compare two graphs for similarity has been of especial interest in the field of pattern recognition (Neuhaus and Bunke 2004). In architecture Conroy and Kirsan (2005) introduced graph matching to analyze genotypic differences in Greek and Turkish Cypriot housing.

The general approach is to define a set of operations to modify graphs and assign each a penalty cost. In order to compare two graphs $G = (V_G, E_G)$ and $H = (V_H, E_H)$ one graph is transformed into the other using these

operations. The cost of the resulting mapping m: $H \rightarrow G$ equals the sum of the costs of its operations. The distance is then derived from the mapping which yields the least penalty cost. This distance metric strongly resembles the Levenshtein distance for string comparison.

In order to maintain relative algorithmic simplicity it was decided to restrict the set of operations to the most basic. Therefore four modification operations are defined (Figure 6), the insertion and the deletion of an edge and the insertion and deletion of a node. The associated costs are c_{IE}, c_{DE}, c_{IN}, c_{DN} respectively and can be adapted at will.

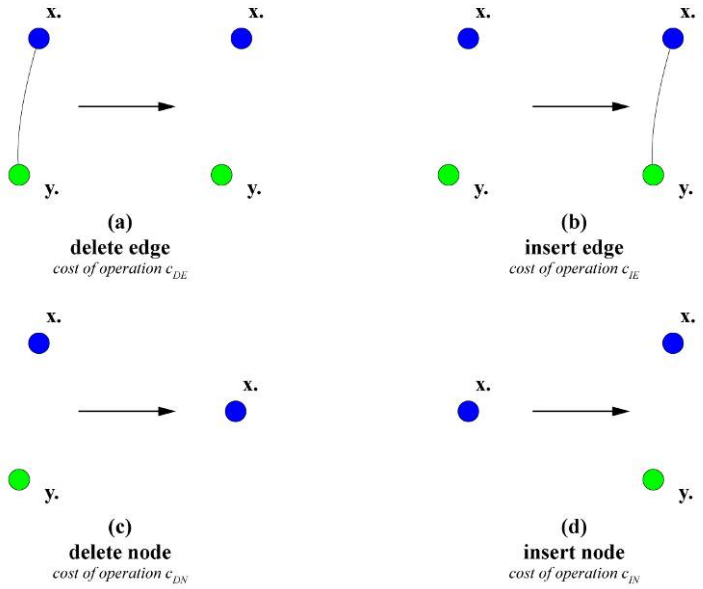

Figure 6. Modification operations; The four basic operations (a) delete edge, (b) insert edge, (c) delete node and (d) insert node.

Further modification operations, such as node re-labeling or complex substitutions, are not included as these are essentially high level operations, which can be replaced by a combination of low level operations, thus once a solution is found the recorded set of operations may be analyzed and replaced by high level operations if required.

2.5. ALGORITHM

The algorithm used to arrive at the edit distance is essentially a graph search algorithm, which returns the shortest path between two nodes. The graph being searched is assembled at run-time; each node represents a mapping m_i: $H \rightarrow G_i$ and each edge stands for a performed modification operation or

a newly mapped node. The start node is the empty mapping m_0: $H \rightarrow H$, while the target node is the correct mapping m_t: $H \rightarrow G$.

Once a node is visited all its unvisited neighbors are examined, their costs *f(x)* are calculated and the nodes are added to a priority queue, which is sorted by the cost *f(x)*. Initially the queue contains only the start node. Subsequently each step examines the first element of the queue until a solution is found with a total cost *f(x)* smaller than or equal to the total cost of the next node in the queue.

Generating the neighbor nodes at run-time requires taking an unmapped node from graph *G*, selecting all possible solutions in *H*, that is all unmapped nodes of equal category, and spawning a new branch for each. Additionally, one branch representing the insertion of a new node is spawned.

In cost based path finding the already covered path from the initial node to the current node is assigned a cost *g(x)*, in the case of graph matching this is the sum of all the cost of the operations used so far

$$g(x) = c_{IN}|V_I| + c_{DN}|V_D| + c_{IE}|E_I| + c_{DE}|E_D| \tag{1}$$

where V_I, V_D, E_I, E_D are the sets of inserted and deleted nodes and edges respectively.

In order to perform efficiently it is desirable to minimize the number of nodes visited by the search; this can best be done by giving the algorithm a sense of direction. Hence, while selecting a node to expand next, nodes closer to the target should be favored over ones still further away. In order to do this a best-case heuristic *h(x)* must be found to estimate the remaining distance to the target mapping. Since a best case heuristic is required one must assume, that all remaining unmapped nodes and edges of *H* can be mapped directly to *G*. Hence, the cost factor lies in the difference between unmapped elements of *H* and *G*, plus the cost of mapping the remaining elements. It follows that

$$\begin{aligned} h(x) &= c_{xN}\left\|V_G| - |V_H| + |V_D| - |V_I\right\| \\ &+ c_{xE}\left\|E_G| - |E_H| + |E_D| - |E_I\right\| \end{aligned} \tag{2}$$

where

$$c_{xN} = \begin{cases} c_{IN}, & \text{if } \left(|V_G| - |V_H| + |V_D| - |V_I|\right) \geq 0 \\ c_{DN}, & \text{otherwise.} \end{cases} \tag{3}$$

$$c_{xE} = \begin{cases} c_{IE}, & \text{if } \left(|E_G| - |E_H| + |E_D| - |E_I|\right) \geq 0 \\ c_{DE}, & \text{otherwise.} \end{cases} \tag{4}$$

Thus the minimal cost function *f(x)* for a node *x* is:

$$f(x) = g(x) + h(x) \tag{5}$$

Combining the cost of the covered path with the estimated cost of the residual path in such a way is known as the A* algorithm (Dechter and Pearl 1985).

3. Application

Part of a GSA funded project is the systematic analysis of federal courthouses. Within this larger framework graph distance analysis was implemented as a tool to describe similarities and eventually arrive at a set of courthouse typologies.

While the work is still in progress and additionally the secure nature of the project further restrict the possibility to depict detailed information some general findings can be shared.

3.1. COURTHOUSE SPECIFIC SETTINGS

The complete area classification table for federal Courthouses implemented in this work is based on the integration of various existing sources such as the courthouse design guide (GSA 1997). For the purpose of this analysis the federal Courthouses are organized into four discrete zones with respect to access requirements. The public zone includes all the areas accessible to the general public along with attorneys, clients, witnesses and jurors. The private zone includes all the functions that have a restricted access and are used by particular courthouse users such as judges, jurors, and employees. The secure zone is provided for the movement and holding of defendants in custody; it includes horizontal and vertical secure circulation systems as well as holding areas. The interface zone comprises the courtroom and its associated areas; it is here that the public, private, and secure zones interact.

One additional zone is defined as carrier space; it essentially represents all exterior spaces and is used to define entrances, windows and orientations.

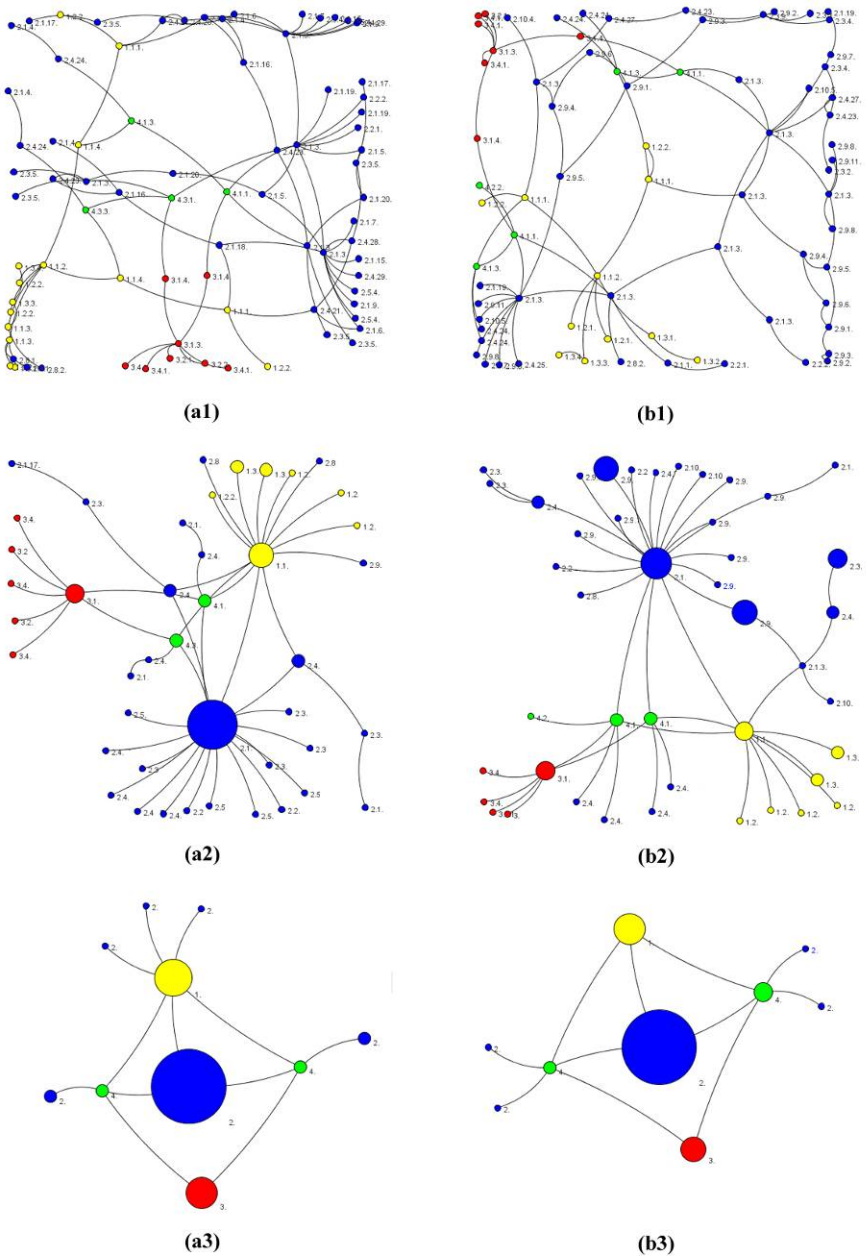

Figure 7. Two courtroom floors with approx. 80 nodes each; (a1) and (b1) show the un-collapsed graphs; (a2) and (b2) are collapsed to level 2; (a3) and (b3) are collapsed to level 1; at this level they have an edit distance of 6

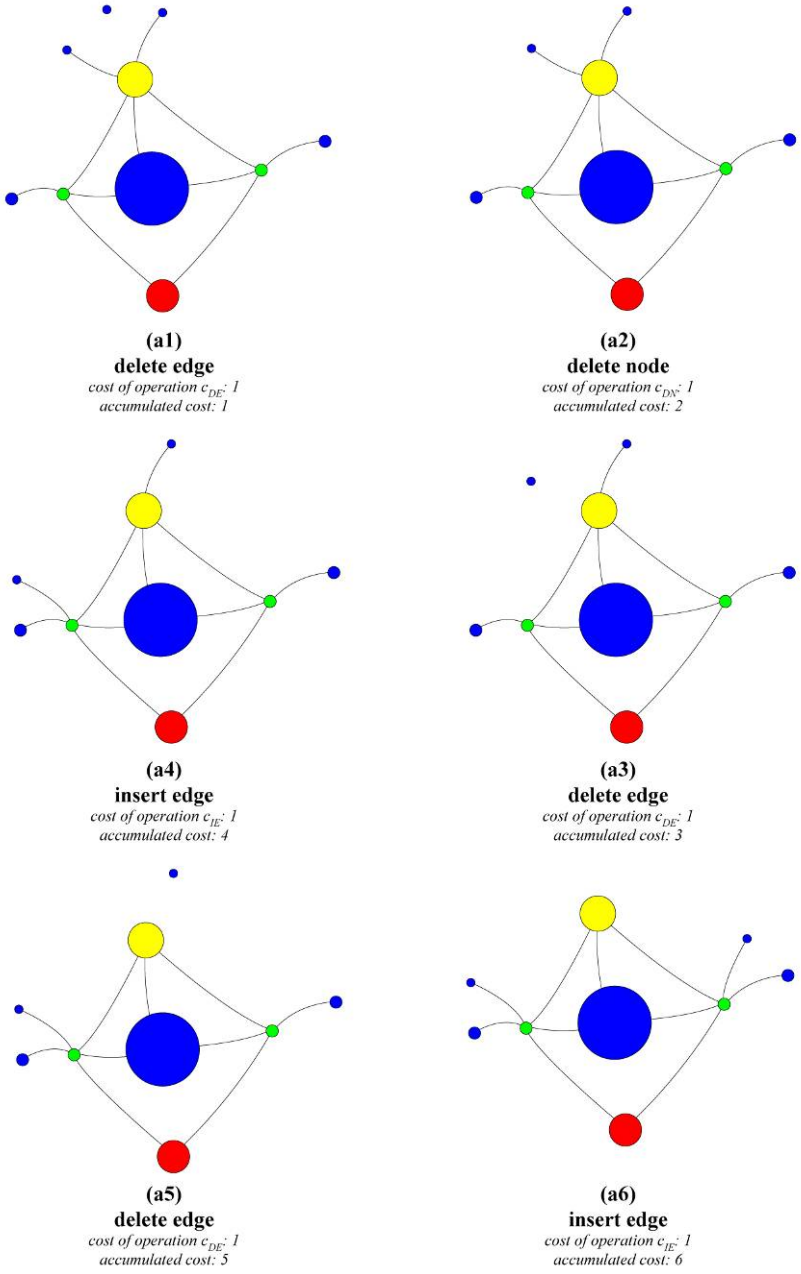

Figure 8. The six steps required to transform the source graph (Figure 7-a3) into the target graph (Figure 7-b3). Since every operation is assigned a cost of 1 the accumulated cost is 6.

Access to secure areas is strictly monitored, these are the areas reserved for the inmates. And finally the courtroom as only interface where agents from all other access levels are brought together is represented as a separate access level.

Area functions are also based on the courthouse design guide (1997) and require two levels in the category tree.

All operations were assigned a uniform cost of 1. Frequently edge operations are assigned a lesser penalty value than node operations; this may sometimes be justified by the nature of the system being abstracted by the graph. In the course of this application it was felt that equal costs best reflect the lean approach taken so far.

3.2. DISCUSSION

When collapsing graphs to the second, intermediate level numerous star configurations occurred (Figures 7-a2 and 7-b2), these are mainly collapsed circulation areas surrounded by other functional areas. A higher level of detail, i.e. more classification levels, and a differentiation of circulation areas by assigning them to the various functional classes would yield more meaningful hyper-nodes.

It is felt that the weight of a hyper-node, the number of collapsed nodes it represents, should somehow be reflected in the distance metric. This could be done by evaluating the difference while mapping a node onto another.

Calculations are extremely taxing on memory; some of the un-collapsed floor plans could not be compared at all. If these calculations remain incomputable despite future improvements, some form of approximation or at least of defining boundaries must be devised.

In addition to making distance measurements feasible, collapsed graph representations seem to also give a good visual representation of the essence of a floor plan. An algorithm to lay out the graph, while supporting this aspect would be helpful, since momentarily the nodes must frequently be repositioned manually.

4. Future Work

There is still ample room for optimizing the algorithm. Additional methods of minimizing the search area must be implemented. One could for example, favor categories with less member-nodes over categories with more when selecting the next node to map, this will reduce the initial complexity of the search tree and reduce the branches spawned before a solution is found.

For large graphs approximations based on the coarser results could be devised, perhaps by recursively calculating the distance of graphs within the hyper-nodes.

Finally, there is still additional need for the construction of architectural meaning emerging from these computations. The major thesis of this work, that graph distance metrics can provide additional insight into the essential structure of given typologies of buildings, will only be tested once these computations with graphs align with computations with spaces to cast light on the study of architectural structure.

Acknowledgements

This research was supported by the General Service Administration (GSA), more specifically by the Office of the Chief Architect. We would like to particularly thank the Center for Courthouse Programs for their invaluable help and architectural insight on the features and complexities of these buildings.

References

Administrative Office of the U.S. Courts: 1997, *U.S. Courts Design Guide*.

de Berg, M, van Kreveld, M, Overmars, M and Schwarzkopf, O: 2000, *Computational Geometry. Algorithms and Applications*, Springer, Berlin.

Bunke, H and Allermann, G: 1983, Inexact graph matching for structural pattern recognition, *Pattern Recognition Letters* **1**(4): 245-253.

Conroy, R and Kirsan, C: 2005, Graph Isomorphism and Genotypical Houses, *in* A van Nes (ed) *Proceedings of the 5th Space Syntax Symposium Vol. 2*, Techne Press, Delft, pp. 15-28.

Dechter, R and Pearl, J: 1985, Generalized best-first search strategies and the optimality of A*, *Journal of the ACM* **32**(3): 505-536.

Hillier, B, Hanson, J and Peponis, J: 1984, What do we mean by building function?, *in* J Powell and I Cooper (eds), *Designing for building utilization*, Spon, London, pp. 61-72.

Karypis, G and Kumar, V: 1995, Analysis of Multilevel Graph Partitioning, *Proceedings of the IEEE/ACM SC95 Conference*, p. 29.

March, L and Steadman, P: 1974, *The Geometry of Environment: An Introduction to Spatial Organization in Design,* The MIT Press, Cambridge.

Neuhaus, M and Bunke, H: 2004, An Error-Tolerant Approximate Matching Algorithm for Attributed Planar Graphs and Its Application to Fingerprint Classification, *in* A Fred, T Caelli, RPW Duin, A Campilho, and D de Ridder (eds), *Structural, Syntactic, and Statistical Pattern Recognition, Proc. Joint IAPR Int. Workshops SSPR and SPR*, Springer, Berlin, pp. 180–189.

Steadman, P: 1976, Graph-theoretic representation of architectural arrangement, *in* L March (eds), *The Architecture of Form*, Cambridge University Press, Cambridge, pp. 94-115.

Stiny, G and Mitchell, W: 1978, The Palladian Grammar, *Environment and Planning B: Planning and Design* **5**: 5-18.