Towards Voxel-Based Algorithms for Building Performance Simulation

Rhys Goldstein, Simon Breslav, and Azam Khan
Autodesk Research, Toronto, ON, Canada

Abstract

This paper explores the design, coupling, and application of algorithms relying on grid-based or voxel-based representations of buildings. Such algorithms may encourage the adoption of building performance simulation in design practice by avoiding the time-consuming and somewhat arbitrary process of manually partitioning a building's interior into polyhedral spaces. They also support detailed visualizations that may open up new possibilities for design tools. This paper presents two grid-based methods: a hierarchical pathfinding algorithm to support detailed simulations of individual occupants, and a very simple heat transfer algorithm to motivate future alternatives to zone models. The coupling of the pathfinding and heat transfer algorithms demonstrates the need for a simulation framework supporting different time advancement patterns. The coupled model is applied to simulate a building with individual occupants that move between rooms and open windows in response to temperature changes.

1 Introduction

The transformation of a building model from an architectural representation to a valid simulation-oriented representation is a source of great frustration for both researchers and practitioners of building performance simulation (BPS). According to Yan et al. (2013), "A lack of interoperability between architecture models and building energy models prevents the efficient use of simulation in the building design process." Similar views were expressed two years prior by Hitchcock & Wong (2011): "Automated data exchange between commonly used software tools for building design and energy analysis remains an elusive goal."

Efforts to improve the architecture-simulation model transformation process can be grouped into three main strategies. The first assumes the use of a traditional energy modeling approach in which a building's interior is manually partitioned into polyhedral spaces. The idea is to improve the quality of an automatically generated simulation model by establishing guidelines for the authoring of a building information model (BIM). Maile et al. (2013) list one such set of guidelines, emphasizing the error-free definition of space boundaries as a key requirement. The second strategy strives to accommodate designers who lack either the expertise or the willingness to define every space with no gaps and no overlapping regions. The idea is to generate spaces automatically from walls, slabs, and other physical building elements. Recent examples of automatic space generation can be found in both academia (Jones et al. 2013) and industry (Molloy 2013). The third strategy avoids both detailed modeling guidelines and sophisticated model transformation tools. The idea is to adopt or invent simulation algorithms which simply do not require space boundaries as input. This paper focuses on the third strategy. Specifically, it explores the replacement of traditional BPS methods with voxel-based algorithms.

A *voxel-based algorithm* operates on a *voxel model*, one of five fundamental ways of representing building geometry as listed in Meng et al. (2007). For the sake of comparison, let us consider the other four representations. A *parametric description* is based on a set of primitives such as rectangular prisms, cylinders, and spheres. Each primitive has parameters such as position, orientation, length, width, height, and radius. BIMs are generally based on parametric descriptions. *Constructive solid geometry* emphasizes the construction of polyhedra through Boolean operations such as union and intersection. This representation relates to traditional building energy modeling, in which the union of all thermal mass volumes ideally fills the entire building (i.e. there are no gaps), and the intersection of any pair is ideally empty (i.e. there are no overlapping regions). *Boundary representations* encode only the surfaces of every physical element, often with a mesh of triangles. Such models are used in typical ray-tracing algorithms for daylight simulation. *Solid representations* specify physical volumes of mass, often with a mesh of tetrahedra. These models are used in finite element methods for structural analysis.

A key advantage of a voxel model is its simplicity when compared with the four alternatives. It can be regarded as a digital image with an extra dimension; instead of a 2D array of squares called *pixels*, one has a 3D array of cubes called *voxels*. By associating each voxel with one or more attributes, such as a material type or density, a voxel model can represent a wide range of 3D objects and their associated properties (Kaufman et al. 1993). Figure 1 illustrates the representation of building geometry with a voxel model. Note that the quality of the model could be improved by refining its resolution, decreasing the size of each voxel and increasing the number of them.

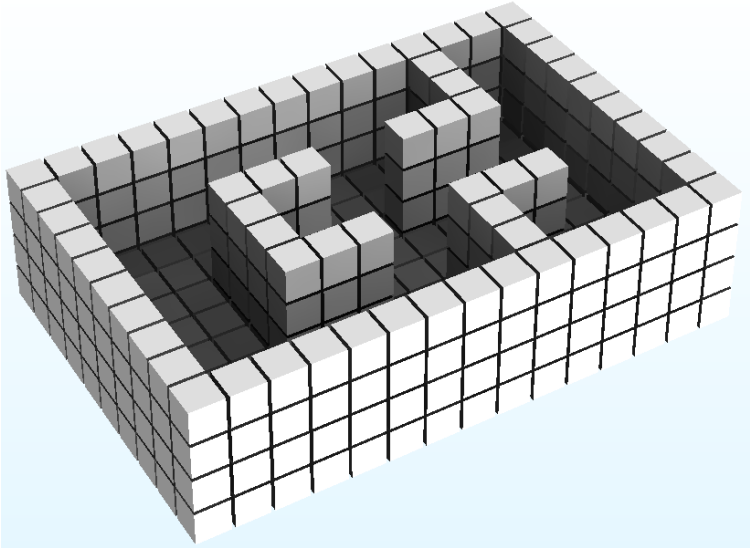

Figure 1: Coarse-resolution voxel model of building geometry

Voxel-based algorithms have been applied to a wide range of domains related to BPS. A classic example is the solving of a temperature field on each point of a grid, as described by Patankar (1980). Although the propagation of light is generally simulated using boundary representations, Levoy (1990) demonstrates that voxel models offer a viable alternative. Savioja et al. (1994) simulates room acoustics with a 3D finite difference mesh, which is another name for a voxel model. One should note that the phrase *finite difference* distinguishes a voxel-based representation from a *finite element* mesh, which is a solid representation. Finally, voxel models are the standard representation for incorporating computational fluid dynamics (CFD) into BPS (Beausoleil-Morrison 2000).

Due to their versatility, voxel models could have a unifying effect on algorithm development for simulation-based building design tools. The transformation of a BIM from a parametric description to a voxel model is relatively straightforward to automate, and once implemented the procedure could be reused to provide input data for many different algorithms. Voxel-based algorithms are particularly compelling for buildings with complex geometry that would be difficult to partition into polyhedral spaces. Buildings with large atria are an example of design projects likely to involve complex geometry, and likely to benefit from the simulation of thermodynamics (Hussain 2012), acoustics (Mahdavi et al. 2007), and other phenomena.

In this paper we present two non-traditional voxel-based BPS algorithms: a pathfinding algorithm for the detailed simulation of occupants, and a heat transfer algorithm that allows temperature to vary nearly continuously over space. The pathfinding algorithm is based on standard shortest path methods, but includes a performance optimization involving the grouping of nearby waypoints to avoid redundant path computations. We consider our pathfinding algorithm adequate for BPS purposes in terms of computational efficiency and quality of the resulting paths. Our heat transfer algorithm has been not validated and may well be overly simplistic for real-world engineering applications. However, it illustrates an alternative approach to energy simulation that avoids the need for space boundaries. Further developments may lead to more accurate voxel-based heat transfer algorithms with significant practical advantages over current methods. After presenting the algorithms independently, we explain how they can be coupled together using a simulation framework supporting non-uniform and asynchronous time advancement patterns. Finally, we present simulation results for a hypothetical hotel with simulated guests and employees who move between rooms and open windows in response to temperature changes.

The algorithms described in this paper are two-dimensional, and might therefore be considered *grid-based* as opposed to *voxel-based*. Some grid-based algorithms are trivial to extend to 3D, while others are more difficult. Regardless, the ultimate goal is a set of 3D voxel-based BPS algorithms that seamlessly input even the most complex examples of building geometry. The work presented here is a step in this direction.

2 Pathfinding Algorithm for Occupant Simulation

Numerous studies have raised concerns over the validity of the simple occupancy models used in practice. In one recent example, the standard ASHRAE 90.2 model is shown to oversimplify manual thermostat adjustment behavior. Consequently, it underestimated winter heating loads in a sample of 82 residential units (Urban & Gomez 2013). More detailed occupant models offer the potential to improve the validity of BPS results, as they have been shown to significantly alter energy use predictions (Hoes et al. 2009). These models may also extend the capabilities of building simulation to new aspects of design. For example, Bourgeois (2005) demonstrate how models which distinguish between individual occupants can be used to predict the performance of both manual and automated lighting control systems.

Among the most detailed occupant models for BPS are those which account for the paths travelled by individual occupants through a building. An example is the USSU system by Tabak (2008). The incorporation of pathfinding enables visualizations that capture the movements of simulated occupants, allowing a modeller to perform a quick yet important visual inspection of simulated behavior. Pathfinding may also compel designers to initiate the BPS modeling process by supplying occupant simulation parameters to predict circulation patterns and analyze evacuation scenarios.

Two classes of building representations are commonly used for pathfinding: meshes and grids. A mesh is a set of points, line segments connecting at least two points, and possibly

surface patches bounded by at least three line segments. There are different methods for transforming a multi-storey building model into a mesh, and different algorithms for exploiting a mesh for occupant simulation. In Whiting et al. (2007), a first set of points is placed at the center of every *portal*, which may be a doorway, ramp, staircase, or elevator. Accessible space is then tessellated into triangles, providing additional points for maneuvering around walls and other obstacles. When travelling between two rooms in a building, simulated occupants select the shortest path under the constraint that they remain on a set of line segments connecting the points. This type of approach is extended in van Toll et al. (2011) to include walkable areas instead of walkable line segments. The resulting representation, where walkable areas are composed of surface patches, is called a *navigation mesh*. An example of a building design tool featuring mesh-based building representations for pathfinding can be found in Doherty et al. (2012).

The other representation used for pathfinding is a grid of cells, which is essentially a 2D version of a voxel model. A simulated individual occupies exactly one grid cell at any point in time. The individual travels through a building by repeatedly moving either orthogonally or diagonally to an adjacent cell while avoiding cells marked as part of an obstacle. A key advantage of these pathfinding methods is that grid-based building representations are generally simpler and more easily generated than mesh-based alternatives. For this reason, most research efforts in grid-based pathfinding assume the grid is given. The objective is to find the shortest path between pairs of cells designated as waypoints.

A simple and efficient method for finding the shortest path between two waypoints on a grid is known as the *A-star* algorithm (Hart et al. 1968). When there are more than two waypoints it is possible to perform an A-star search on every pair, as was done to simulate building occupants in Breslav et al. (2013). However, the computational effort required for this brute force approach increases with the square of the number of waypoints in the building. Given a high-resolution model of a large building, where every workstation, every appliance, and every spot at a dining table has its own waypoint, the time required to compute the shortest path for every pair of waypoints may dissuade designers from adopting detailed occupant simulation as a design aid. Fortunately, there are ways to speed up grid-based pathfinding, particularly if the paths need not be absolutely optimal. For example, Botea et al. (2004) propose a hierarchical approach in which pathfinding information is precomputed for non-overlapping subregions. In their example, a 40-by-40-cell grid is decomposed into sixteen 10-by-10-cell subregions.

Here we present a grid-based pathfinding algorithm similar to Botea et al. (2004) in that it uses a hierarchy to reuse pathfinding information, but different in that the hierarchy groups nearby waypoints instead of nearby cells. The key concept is that if two waypoints are close together, the shortest path towards one waypoint is likely to be a reasonably short path towards the other, at least while the occupant is at a considerable distance. It is only when an occupant gets close that the shortest paths toward the two waypoints are likely to diverge. Our method therefore precomputes paths to both waypoints over a small surrounding region; beyond this region, paths to only one of the two waypoints are precomputed.

To better illustrate the pathfinding algorithm, consider the example shown on the left of Figure 2. A simple, hypothetical building storey is represented by a 15-by-10 grid of cells. The objective is to precompute a path from every walkable cell (non-grey) to each of three waypoints (A, B, and C), where the paths may not enter cells marked as obstacles (grey). The first step is to calculate the shortest paths over separate search regions which gradually expand around each waypoint. This is done with a routine application of Dijkstra's shortest path algorithm, which calculates the distance from all nodes of a mathematical graph to any particular node. In the case of a grid, each cell is a node. To avoid round-off errors, we associate

a dimensionless cost of 5 with travel to an adjacent cell in the four directions orthogonal to the grid axes, and a cost of 7 with travel to a diagonally adjacent cell. The search regions expand until any two of them come into contact with one another, as shown on the right of Figure 2.

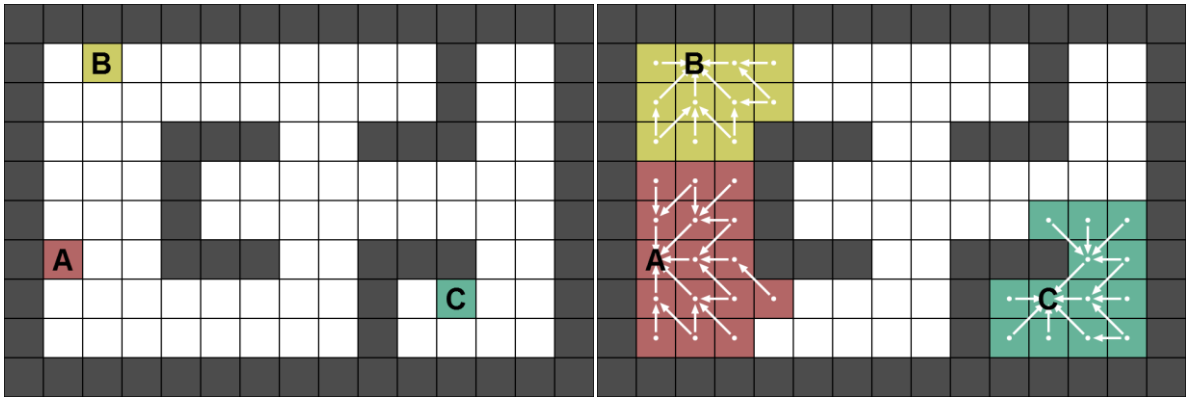

Figure 2: Pathfinding example: initial expansion of search regions

In the example, the search regions surrounding waypoints A and B were the first to collide. To reduce the computation time and memory use, the waypoints become grouped before all search regions are allowed to continue expanding. As part of the grouping process, one waypoint is arbitrary assigned to be the parent of the other. In this example, waypoint B becomes the parent. Because waypoint A is the child, its search region is expanded independently until it completely engulfs the search region of waypoint B. Waypoint A's final search region, shown in Figure 3, is then stored for use during the simulation. Any occupant heading to A will follow one of the paths indicated by the arrows, provided that they are within this region. An occupant that is further away will first head towards the parent, waypoint B, until they enter the search region of A.

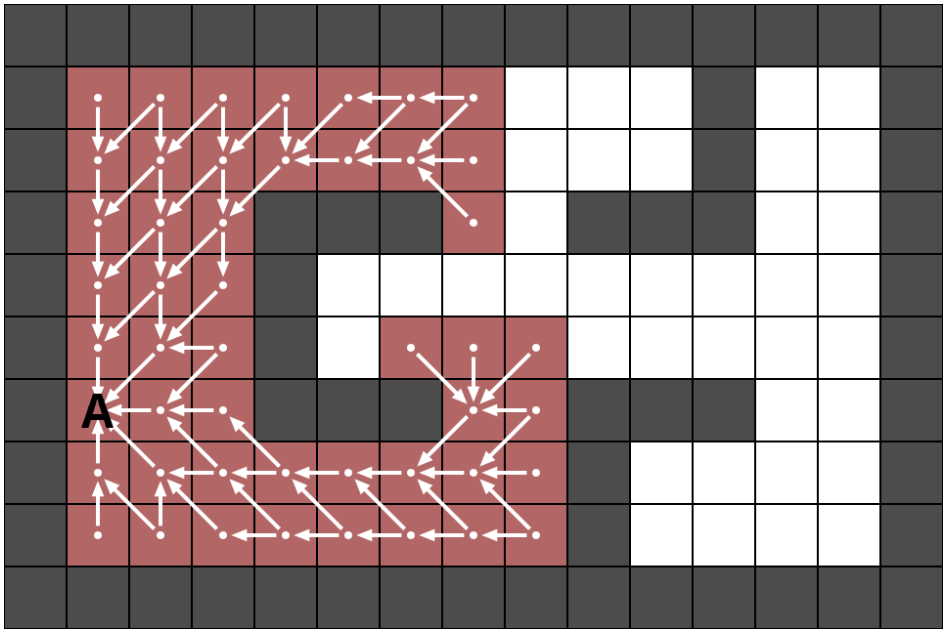

Figure 3: Pathfinding example: final search region for waypoint A

With the precomputation complete for waypoint A, pathfinding resumes for B and C. Initially, the search region of B is extended to fill the search region of A at the time of the collision (i.e. on the right Figure 2). Eventually the search regions of B and C will collide. At this point, most of the entire building storey will have been covered, and when the search regions are independently expanded they will both include every walkable cell. Figure 4 shows the final search regions for B and C, which include the shortest paths from anywhere in the building. This completes the precomputation of paths.

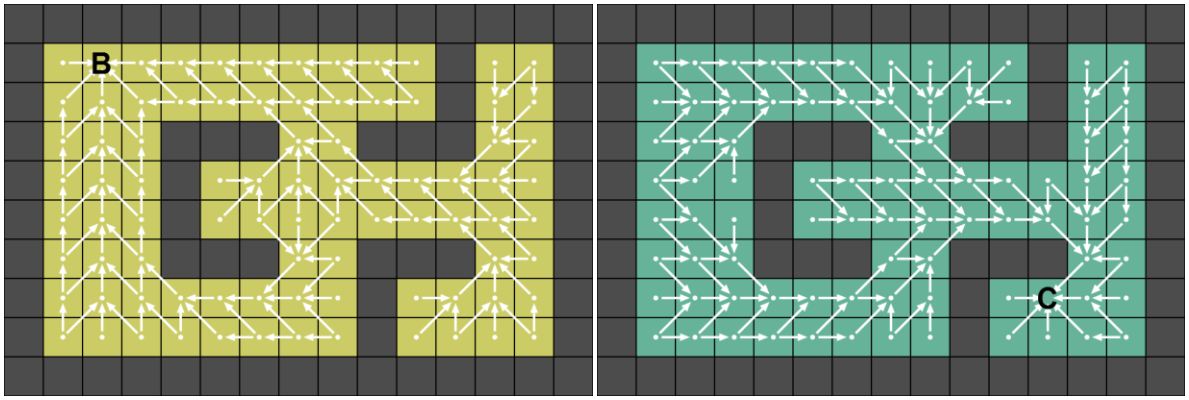

Figure 4: Pathfinding example: final search regions for waypoints B and C

During a simulation, suppose that an occupant decides to move from C to A. The occupant begins outside of waypoint A's search region, but within the search region of waypoint B, the parent of A. As shown in Figure 5, the occupant initially moves towards B, but enters the search region of A on route and changes course when the paths diverge.

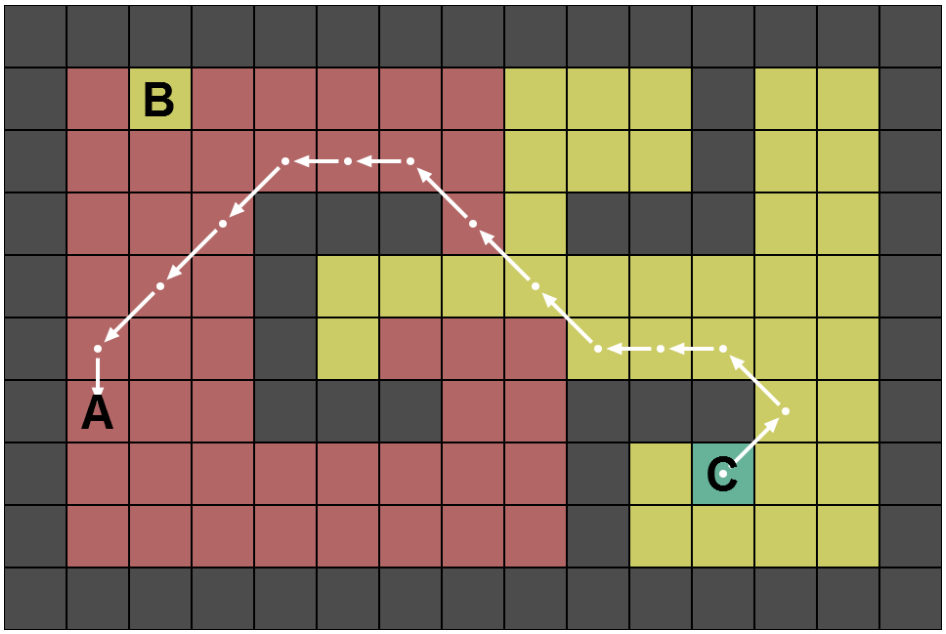

Figure 5: Pathfinding example: computed path from waypoint C to A

The example illustrates the approximate nature of the algorithm, as the resulting path from C to A is reasonable but not optimal. Had the search region of A been expanded one cell further, the occupant would have travelled a slightly shorter distance by passing the c-shaped

obstacle towards the bottom of Figure 5 instead of the top. The fact that a brute force approach would have yielded the correct path might lead one to question the value of the proposed algorithm. After all, the only benefit in this case was the fact that the search region of waypoint A was reduced by a factor of two. However, given a large building model with hundreds of waypoints, the savings in computation time and memory would be considerable. Also, the example above was contrived to be a worst-case scenario. While future work is needed to quantify the error rate, observations on a larger and more realistic model have convinced us that the algorithm will rarely yield a suboptimal path.

Note that the grouping of waypoints is recursive in nature. Waypoint Q may first become a parent of waypoint D, and later a child of waypoint Z. An occupant travelling to waypoint D may begin by heading towards waypoint K, followed by Z and later Q. Despite being guided by a succession of search regions, it is possible if not probable that the occupant's overall path will be optimal.

As presented, the algorithm is a grid-based method that is relatively straightforward to extend to multiple storeys, assuming each storey can be treated as flat. A truly voxel-based adaption would be needed for irregular buildings such as the Guggenheim Museum in Manhattan, which features a tall central atrium with a continuously spiralling ramp. If the ramp were represented in a voxel model, the discretization of space would produce numerous single-voxel steps. Simulated occupants would therefore be permitted to ascend or descend a single voxel, but a two-voxel step would be considered a barrier to normal travel. The development of a voxel-based version of our pathfinding algorithm would be a natural next step towards voxel-based BPS, and it would likely be far simpler than a mesh-based alternative.

3 Heat Transfer Algorithm Avoiding Space Boundaries

Zone models have a long tradition as the primary means of predicting heat flow in BPS. As explained in Spitler (2011), there are many types of zone models. They include the highly detailed network models described by Clarke (2001), as well as simpler alternatives, but they all use similar building representations in which interior space is partitioned into polyhedra referred to as *spaces*. Each space represents a thermal mass that is assumed to have uniform temperature at every point in simulated time. Elements that bound spaces, such as walls and windows, are also partitioned into thermal masses.

In addition to the inconvenience of identifying space boundaries, which was mentioned at the outset, the use of zone models for heat transfer may involve a number of arbitrary decisions in cases where a connected interior region must be partitioned into multiple spaces. In Figure 6, the same building geometry used to demonstrate pathfinding is shown decomposed into alternative zone models.

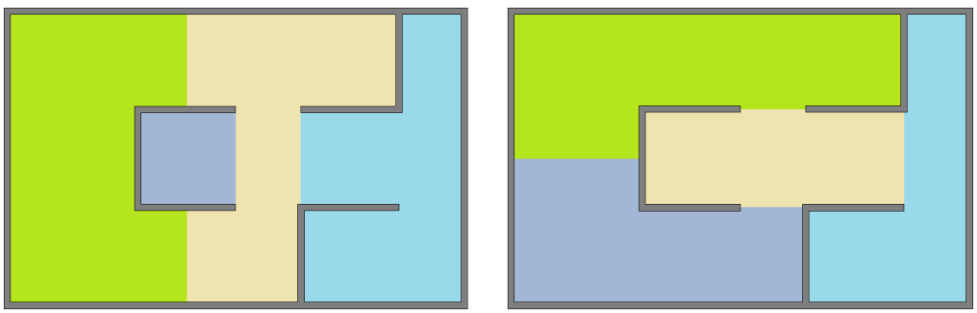

Figure 6: Two different zone models for identical building geometry

Both models in Figure 6 have four spaces. However, as the boundaries are completely different, the resulting temperature predictions will differ. This is one of the reasons why different energy modellers can expect to obtain different BPS results even when given the same architectural model, the same weather data, and the same operating conditions. Recent developments to automate the generation of space boundaries may help standardize the energy modeling process. We argue that alternative algorithms, which may eliminate the need for polyhedral spaces, also deserve consideration as a means of addressing the cumbersome and arbitrary nature of zone model preparation. To illustrate, we present a simple heat transfer algorithm. It is defined for two dimensions, but we note that it could be extended to 3D.

In 2D, each building storey is represented as a grid of cells. A cell at coordinates (i, j) has a thermal resistance value R_{ij} and a temperature T_{ij}. As the algorithm is illustrative in nature, we are not prepared to give a formula for R_{ij} based on material properties and the geometry of building elements. It is quite possible that a valid voxel-based heat transfer algorithm would require several thermal parameters associated with each cell, and that these parameters would have to be adjusted to compensate for geometric errors introduced when building elements are discretized into voxels. In the algorithm presented here, R_{ij} is simply given a small value for cells representing air, a larger value for windows, and a still larger value for walls. The larger the value of R_{ij}, the longer it takes for cell (i, j) to exchange heat with its neighbors. We use the formula below to update T_{ij} at every cell for every time step.

$$T_{ij}' = c_{ij(i-1)j} \cdot T_{(i-1)j} + c_{iji(j-1)} \cdot T_{i(j-1)} + c_{ij(i+1)j} \cdot T_{(i+1)j} + c_{iji(j+1)} \cdot T_{i(j+1)} + (1 - c_{ij(i-1)j} - c_{iji(j-1)} - c_{ij(i+1)j} - c_{iji(j+1)}) \cdot T_{ij}$$

where

$$c_{ij(i-1)j} = \frac{1}{4 \cdot \sqrt{R_{ij} \cdot R_{(i-1)j} + 1}} \qquad c_{iji(j-1)} = \frac{1}{4 \cdot \sqrt{R_{ij} \cdot R_{i(j-1)} + 1}}$$

$$c_{ij(i+1)j} = \frac{1}{4 \cdot \sqrt{R_{ij} \cdot R_{(i+1)j} + 1}} \qquad c_{iji(j+1)} = \frac{1}{4 \cdot \sqrt{R_{ij} \cdot R_{i(j+1)} + 1}}$$

Voxel-based heat transfer algorithms such as the one presented allow temperature to vary in a nearly continuous manner down corridors and across rooms. Their primary advantage, however, is that it is far simpler to transform an architectural model into a voxel model than a zone model. Whereas the specification of arbitrary space boundaries may require numerous parameters, the creation of a voxel model may require only one: the voxel size. The voxel size parameter may actually be considered an asset, as it gives designers a simple mechanism for controlling the trade-off between computation time and accuracy.

Ultimately, it will be essential to quantify the relationship between computation time, accuracy, and voxel size. If is it found that relatively coarse voxel models can produce sufficiently accurate heat transfer predictions, these simulations may be fast enough for design applications. One reason to be optimistic is that, due to decades of advances in computer chips and more recent investments in cloud computing, designers and engineers have access to considerably more computing power now than when zone models were first adopted for BPS.

Although we are not yet prepared to recommend our voxel-based heat transfer algorithm or any other for serious quantitative use in BPS practice, we can point to a number of research efforts that indicate a recent interest in finding simpler methods for heat transfer simulation. The most obvious example is zone reduction, as investigated by Dobbs & Hencey (2003) and Korolija & Zhang (2013), where multiple spaces are grouped into a single thermal mass. Simplified models have also been developed using statistical methods (Melo et al.

2013). All of these works compare results obtained with those of traditional methods, as would have to be done for any voxel-based approach prior to adoption.

It is worth noting that while the presented algorithm does not solve fluid flow, CFD methods are traditionally voxel-based. Furthermore, a number of very simple CFD algorithms exist. The solver in Stam (2003), for example, was implemented in about 100 lines of code. It includes a diffusion step that is somewhat similar to the heat transfer algorithm presented here, though it also accounts for advection and conservation of mass. Although Stam's CFD algorithm was developed for the entertainment industry, the idea of applying it to structural optimization for building design was explored in Chronis et al. (2011). The adoption of a fast CFD solver for whole-building airflow simulation would become increasingly compelling if other voxel-based algorithms were introduced into BPS practice.

4 Coupling Voxel-Based Algorithms

A transition towards voxel-based BPS algorithms would introduce regularity into the representation of building geometry, as polyhedra are replaced with cubes of uniform size and shape. Interestingly, voxel-based algorithms may introduce variability into the representation of time. Whereas transitions occur at regular intervals in many BPS algorithms, variable time steps may become commonplace if voxel-based algorithms are adopted.

Consider the pathfinding algorithm presented above, and suppose that it takes 5 seconds for an occupant to move one cell in a direction orthogonal to the grid axis. It must then take 7 seconds for the occupant to move diagonally. Thus, certain time steps will be 5 seconds in duration, and others will be 7. However, since there could be multiple occupants moving simultaneously in different parts of the building, the time remaining before the most imminent 1-cell move could be as little as 1 second. For example, if one occupant moved three times in an orthogonal direction (15 seconds), they would complete their trip 1 second after another occupant moved twice in a diagonal direction (14 seconds).

Grid-based or voxel-based occupant movement is an example of discrete-event simulation, where future events are scheduled irrespective of any time step. By contrast, the heat transfer algorithm we presented is intended to be a discrete time simulation, where transitions occur at a fixed time step. The problem arises of how to couple simulation algorithms with different time advancement patterns. One solution is to implement each algorithm according to the modeling conventions known as the Discrete Event System Specification (DEVS).

The key concept of DEVS theory, described in Zeigler et al. (2000), is that essentially any simulation can be formulated using external and internal events which occur at any point in time. Both types of events depend on the elapsed time since the previous event, and result in a remaining time before the next internal event. An external event also depends on the incoming message that triggers it, whereas an internal event has the option of producing outgoing messages. Multiple DEVS models, each with their own external and internal event functions, can be coupled together to communicate asynchronously via message passing.

To couple our grid-based pathfinding and heat transfer algorithms, we first define a separate DEVS model for each. The pathfinding is implemented as part of the Behavior model, which schedules a future event for every occupant's next movement from one cell to another. The remaining time is always the time before the most imminent arrival, which is a highly non-uniform duration. Whenever this duration elapses, an internal event occurs and produces an outgoing message indicating which occupant moved and to which cell. The Thermal model, by contrast, simply outputs an array at regular time intervals. The array gives the temperature at every cell location in the building.

The next step is to define a Comfort model to receive the messages from both the Behavior and Thermal models. Whenever the Comfort model receives a message from the Be-

havior model indicating that an occupant moved, it replies with a message indicating the new temperature of the occupant's surroundings based on the previously received temperature array. Whenever the Comfort model receives a new temperature array from the Thermal model, it sends numerous messages to the Behavior model indicating the new temperature of every occupant's surroundings. More realistic comfort models can be implemented in this fashion, but the key concept is that pathfinding and heat transfer algorithms are coupled despite having completely different time advancement patterns.

More information on using DEVS for BPS can be found in Goldstein et al. (2013). The formalism has advantages other than providing a message passing framework as described above. Gunay et al. (2014) argues that stochastic models predicting occupant actions are best formulated with variable time steps, as supported by DEVS. Wang el al. (2013) present a building evacuation simulation to illustrate how modellers can take advantage of web-based computing infrastructure specifically developed for arbitrary DEVS models.

5 Illustrative Results

We tested the coupled grid-based pathfinding and heat transfer algorithms using the hypothetical two-storey hotel model shown in Figure 7.

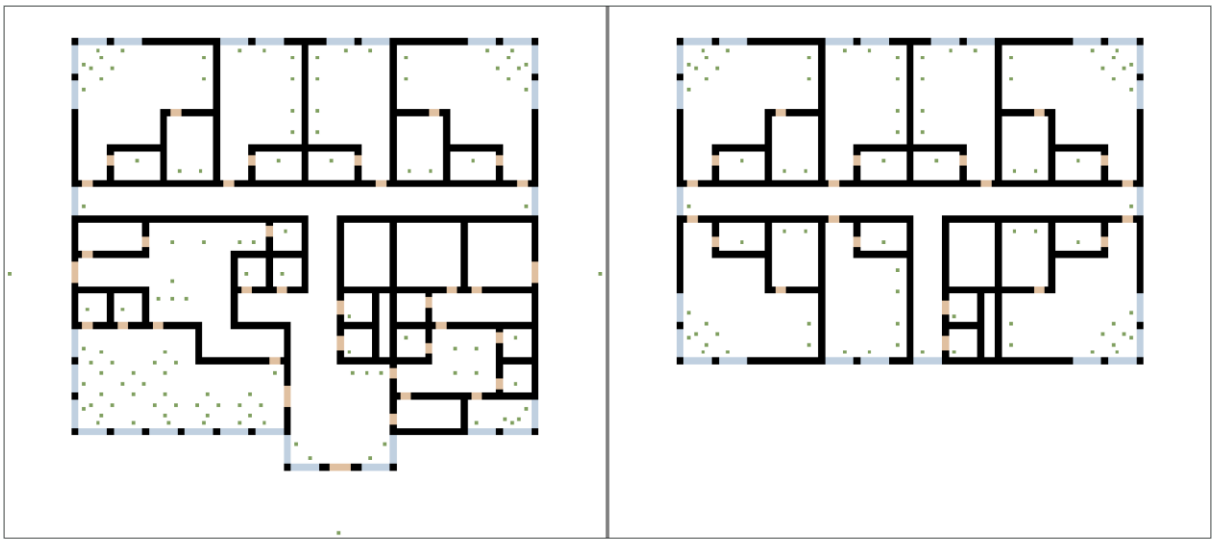

Figure 7: Grid-based floor plan of a hypothetical hotel

The main entrance is towards the bottom of the first storey on the left of Figure 7. Doors (beige strips) lead into the building, to the restaurant on the left, to the hotel office on the right, and into the elevators straight ahead. Corridors provide access to the guest rooms towards the rear of the first storey and throughout the second storey on the right of Figure 7. Waypoints (green dots) represent places where guests eat and sleep, places where hotel employees prepare food and perform office work, washrooms, elevators, and offsite locations. There is also a waypoint in front of the windows (blue strips), as an occupant may choose to open a window to cool the indoor air.

It should be intuitive from the number of waypoints that restricting their search regions in the pathfinding process could greatly reduce computation time and memory use. We implemented the algorithm in an interpreted language (Lua), yet still found that the entire pathfinding precomputation would complete in a matter of seconds.

The results of voxel-based algorithms are conducive to advanced 3D visualization techniques, including some of those demonstrated in Hailemariam et al. (2010). In Figure 8,

cylindrical glyphs are used to indicate the locations of guests (green) and hotel employees (purple). Subtle trails are animated behind moving occupants, revealing the routes computed by the pathfinding algorithm. Although the paths are not guaranteed to be optimal, we did not observe any cases in which it was obvious that a travelled route could have been shortened.

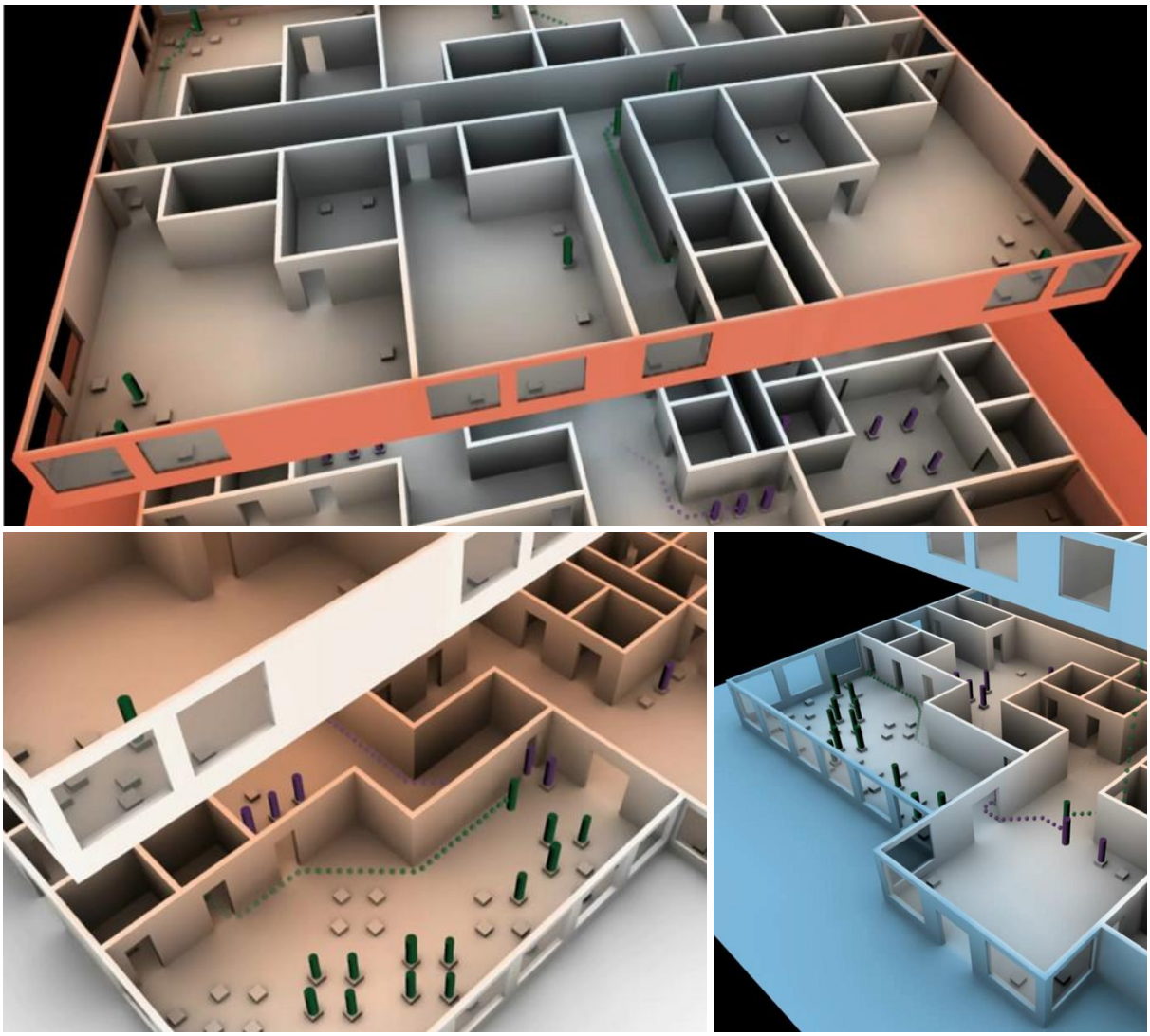

Figure 8: 3D Visualization of simulated occupants and temperature

Color representing temperature is overlaid on the walls and floor slabs to show the results of the heat transfer algorithm. The image at the top of Figure 8 shows the hotel in the late morning when the indoor air is warming up from the outside inwards. Later in the day, as shown in the bottom left image, the outdoor air has dropped and the warm indoor air is cooling. The bottom right image shows evening conditions, when the outermost areas of the building have cooled considerably. By simplifying the representation of spatial data, voxel-based methods may allow such visualizations to be generated automatically. This could provide rapid insights to designers, thereby encouraging them to adopt BPS.

Unlike traditional BPS methods, a grid- or voxel-based heat transfer algorithm allows temperatures to vary smoothly across a room or down a corridor. This effect can be seen in Figure 9, which shows simulated temperatures throughout the hotel at 9:00 pm in simulated time. It is evident by the temperature variations in certain areas that the simulated occupants have opened windows in both the restaurant and several of the guest rooms.

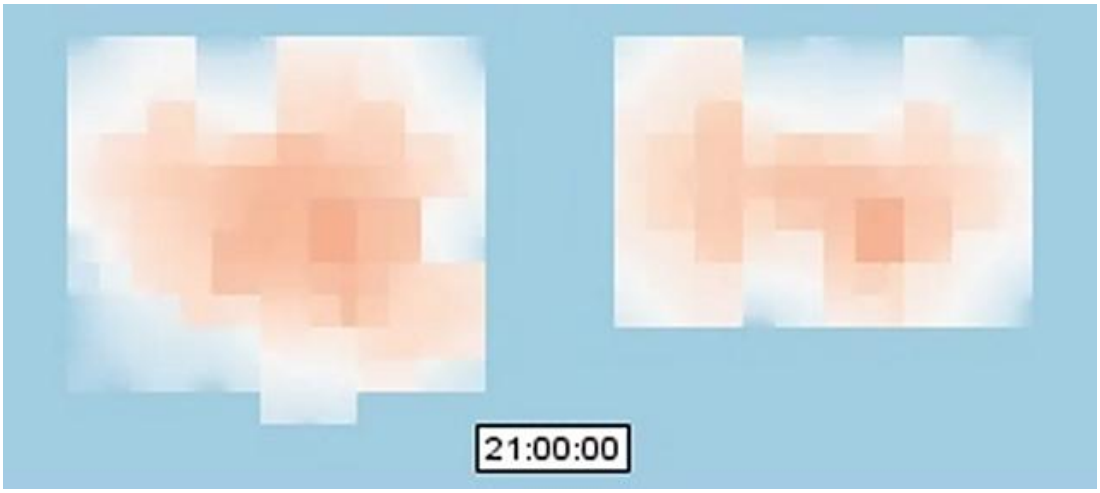

Figure 9: 2D visualization of simulated temperature

Note that neither grid-based nor voxel-based algorithms require walls to be orthogonal to the axes as they are in this hotel example. Off-axis and irregularly shaped walls can be represented in a voxel model in the same way that diagonal lines and curves can be displayed on a pixel-based screen.

6 Conclusions

It has been said that "space is the most important object for BPS" (Maile et al. 2013), referring to the need to manually partition a building's interior into polyhedra while avoiding gaps and overlapping regions. This may be true assuming a traditional approach to energy modeling, but recent developments in automated space generation and future developments in simulation algorithms may relieve BPS users of this burden. Voxel-based algorithms are particularly compelling because they rely on one of the simplest representations of building geometry. Also, looking at a wide range of fields, these algorithms have been applied to simulate a diverse set of phenomena including temperature, light, acoustics, and fluid dynamics. With a vision that a number of BPS methods could be replaced with voxel-based alternatives, we have described a computationally efficient pathfinding algorithm suitable for detailed occupant simulation. The heat transfer algorithm in this paper illustrates what BPS might look like if zone models were replaced with voxel models, though considerable research is needed before such a method can be adopted in practice. The presented simulation of a hotel demonstrates how coupled voxel-based algorithms may produce highly detailed results and make compelling visualizations available for use in the design process.

7 References

Beausoleil-Morrison, I. 2000. *The Adaptive Coupling of Heat and Air Flow Modelling within Dynamic Whole-Building Simulation*. Ph.D. Thesis. University of Strathclyde, Glasgow, UK.

Botea, A., Müller, M. & Schaeffer J. 2004. Near Optimal Hierarchical Path-Finding. In *Game Development*. Vol. 1, pp. 7-28.

Bourgeois, D. 2005. *Detailed Occupancy Prediction, Occupancy-Sensing Control and Advanced Behavioural Modelling Within Whole-Building Energy Simulation*. Ph.D. Thesis. Laval University, Quebec, Canada.

Breslav, S., Goldstein, R., Doherty, B., Rumery, D. & Khan, A. 2013. Simulating the Sensing of Building Occupancy. In *Symposium on Simulation for Architecture and Urban Design*. San Diego, CA, USA.

Chronis, A., Turner, A. & Tsigkari, M. 2011. Generative Fluid Dynamics: Integration of Fast Fluid Dynamics and Genetic Algorithms for Wind Loading Optimization of a Free Form Surface. In *Symposium on Simulation for Architecture and Urban Design*. Boston, MA, USA.

Clarke, J.A. 2001. *Energy Simulation in Building Design*. 2nd Edition. Oxford: Butterworth-Heinemann.

Dobbs, J.R. & Hencey, B.M. 2013. Structured Building Model Reduction Toward Parallel Simulation. In *Building Simulation Conference*. Chambéry, France.

Doherty, B., Rumery, D., Barnes, B. & Zhou, B. 2012. A Spatial Query & Analysis Tool for Architects. In *Symposium on Simulation for Architecture and Urban Design*. Orlando, FL, USA.

Goldstein, R., Breslav, S. & Khan, A. 2013. Using General Modeling Conventions for the Shared Development of Building Performance Simulation Software. In *Building Simulation Conference*. Chambéry, France.

Gunay, B.H., O'Brien, L., Beausoleil-Morrison, I., Goldstein, R., Breslav, S. & Khan, A. 2014 (to appear). Coupling Stochastic Occupant Models to Building Performance Simulation using the Discrete Event System Specification (DEVS) Formalism. Accepted for publication in *Building Performance Simulation*.

Hailemariam, E., Glueck, M., Attar, R., Tessier, A., McCrae, J. & Khan, A. 2010. In *IBPSA-Canada eSim Conference*. Winnipeg, AB, Canada.

Hart, P., Nilsson, N. & Raphael, B. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. In *IEEE Transactions on Systems Science and Cybernetics*. Vol. 4, No. 2, pp. 100–107.

Hitchcock, R.J. & Wong, J. 2011. Transforming IFC Architectural View BIMs for Energy Simulation. In *Building Simulation Conference*. Sydney, Australia.

Hoes, P., Hensen, J.L.M., Loomans, M.G.L.C., de Vries, B. & Bourgeois, D. 2008. User Behavior in Whole Building Simulation. In *Energy & Buildings*. Vol. 41, No. 3, pp. 295-302.

Hussain, S. 2012. *Numerical Investigations of the Indoor Thermal Environment in Atria and of the Buoyancy-Driven Ventilation in a Simple Atrium Building*. Ph.D. Thesis. Queen's University, Kingston, ON, Canada.

Jones, N.L., McCrone, C.J., Walter, B.J., Pratt K.B. & Greenberg, D.P. 2013. Automated Translation and Thermal Zoning of Digital Building Models for Energy Analysis. In *Building Simulation Conference*. Chambéry, France.

Kaufman, A., Cohen, D. & Yagel, R. 1993. Volume Graphics. In *IEEE Computer*. Vol. 26, No. 7, pp. 51-64.

Korolija, I. & Zhang, Y. 2013. Impact of Model Simplification on Energy and Comfort Analysis for Dwellings. In *Building Simulation Conference*. Chambéry, France.

Levoy, M. 1990. Efficient Ray Tracing of Volume Data. In *ACM Transactions on Graphics*. Vol. 9, No. 3, pp. 245-261.

Mahdavi, A., Pak, J., Lechleitner, J. 2007. Acoustics of Atria: Contrasting Measurement and Modeling Results. In *Building Simulation Conference*. Beijing, China.

Maile, T., O'Donnell, J., Bazjanac, V. & Rose, C. 2013. BIM - Geometry Modelling Guidelines for Building Energy Performance Simulation. In *Building Simulation Conference*. Chambéry, France.

Melo, A.P., Lamberts, R., Cóstola, D. & Hensen, J.L.M. 2013. Development of a Method to Predict Building Energy Consumption through an Artificial Neural Network Approach. In *Building Simulation Conference*. Chambéry, France.

Meng, L. & Forberg, A. 2007. 3D Building Generalisation. Chapter 11 in *Challenges in the Portrayal of Geographic Information: Issues of Generalisation and Multi Scale Representation*, Mackaness, W., Ruas, A. & Sarjakoski, T., Eds. Amsterdam: Elsevier Science.

Molloy, Ian. 2013. From BIM to BPA: What is an 'Energy Analysis Model' (EAM). *Building Performance Analysis*. Autodesk Inc. Website (accessed December 11, 2013). http://autodesk.typepad.com/bpa/2013/12/from-bim-to-bpa-what-is-an-energy-analysis-model-eam.html

Patankar, S.V. 1980. *Numerical Heat Transfer and Fluid Flow*. Hemisphere Series on Computational Methods in Mechanics and Thermal Science. Washington, DC: Hemisphere Publishing Corporation.

Savioja, L. 1994. Simulation of room acoustics with a 3-D finite difference mesh. In *International Computer Music Conference*, Aarhus, Denmark.

Spitler, J.D. 2011. Thermal Load and Energy Performance. Chapter 5 in *Building Performance Simulation for Design and Operation*, Hensen, J.L.M. & Lamberts, R., Eds. New York: Spon Press.

Stam, J. 2003. Real-Time Fluid Dynamics for Games. In *Game Developers Conference*. San Jose, CA, USA.

Tabak, V. 2008. *User Simulation of Space Utilisation*. Ph.D. Thesis. Eindhoven University of Technology, Netherlands.

van Toll, W., Cook IV, A.F. & Geraerts, R. 2011. Navigation Meshes for Realistic Multi-Layered Environments. In *IEEE/RSJ International Intelligent Robots and Systems Conference*. San Francisco, CA, USA.

Urban, B. & Gomez, C. 2013. A Case for Thermostat User Models. In *Building Simulation Conference*. Chambéry, France.

Wang, S., Wainer, G., Goldstein, R. & Khan, A. 2013. Solutions for Scalability in Building Information Modeling and Simulation-Based Design. In *Symposium on Simulation for Architecture and Urban Design*. San Diego, CA, USA.

Whiting, E., Battat, J. & Teller, S. 2007. Topology of Urban Enrivonments. In *Computer-Aided Architectural Design Futures (CAADFutures)*, Sydney, Australia.

Yan, W., Clayton, M., Haberl, J., Jeong, W., Kim, J.B., Kota, S., Alcocer, J.L.B. & Dixit, M. 2013. Interfacing BIM with Building Thermal and Daylighting Modeling. In *Building Simulation Conference*. Chambéry, France.

Zeigler, B.P., Praehofer, H. & Kim, T.G. 2000. *Theory of Modeling and Simulation*. 2nd Edition. San Diego: Academic Press.