CRESCIMENTO URBANO, SIMULAÇÕES E ESPAÇOS CELULARES: ESTUDO DE DESEMPENHO DE UM SIMULADOR DE CRESCIMENTO URBANO

M. V. P. Saraiva, O. M. Peres, M. C. Polidori

RESUMO

Cidades em crescimento têm sido assumidas como fenômenos complexos, envolvendo grande quantidade de fatores urbanos, naturais e institucionais que interagem em diferentes escalas. Modelos de computador têm sido utilizados com sucesso para entender esse processo, possibilitando a realização de simulações da dinâmica urbana. Este artigo trata do modelo de simulação de crescimento urbano denominado SACI® - Simulador do Ambiente da Cidade (Polidori, 2004), que realiza simulações de crescimento urbano considerando integradamente fatores urbanos, naturais e institucionais, promovendo simultaneidade entre crescimento externo e interno a um espaço urbano preexistente, representados e modelados utilizando integradamente grafos, autômato celular e geotecnologias. O objetivo deste trabalho é revisar as características conceituais e computacionais do modelo SACI, identificando as atuais limitações e potencialidades do software, de modo a desenvolver uma nova versão do sistema.

1 INTRODUÇÃO

Cidades em crescimento têm sido assumidas como fenômenos complexos, envolvendo grande quantidade de fatores urbanos, naturais e institucionais. Estes fatores interagem em diferentes escalas e mudam ao longo do tempo, desafiando a ciência a encontrar explicações, nexos e padrões. Modelos de computador têm sido utilizados com sucesso para entender o processo de crescimento das cidades, permitindo reproduzir a cidade e a paisagem artificialmente e possibilitando a realização de simulações da dinâmica urbana.

Este artigo trata do modelo de simulação de crescimento urbano denominado SACI® - Simulador do Ambiente da Cidade, desenvolvido por Polidori (2004) e implementado como uma extensão do software ArcView®, utilizando seus recursos nativos de Avenue e a linguagem de programação C++.

A proposta do trabalho é revisar o modelo de simulação de crescimento urbano SACI, suas características conceituais e computacionais, identificando as atuais limitações e potencialidades do software, com o objetivo de desenvolver uma nova versão do sistema.

Os principais problemas identificados pelos usuários do sistema são: 1) excessivo tempo de processamento necessário para efetuar as simulações; 2) dependência de software proprietário de SIG, no caso, o ArcView®, da ESRI. O estudo das soluções para estes problemas tem quatro etapas de trabalho: a) estudar o modelo de simulação de crescimento urbano SACI e sua implementação computacional; b) encontrar, no algoritmo geral do sistema, as rotinas responsáveis pelo desempenho atual do software; c) desenvolver algoritmos mais eficientes ou alternativos para as rotinas encontradas na etapa anterior; d) implementar esses algoritmos em um sistema independente, com suporte aos formatos de arquivos utilizados pelos softwares de SIG mais tradicionais.

2 O MODELO DE SIMULAÇÃO DE CRESCIMENTO URBANO

O modelo de simulação de crescimento urbano SACI® é dedicado a realizar estudos da dinâmica espacial urbana, considerando integradamente fatores naturais, urbanos e institucionais, implementando crescimento interno e externo à cidade preexistente. A cidade e o território são modelados em ambiente computacional, reeditando os modelos de potencial e centralidade (Krafta, 1994), com apoio em teoria de grafos, autômato celular e geotecnologias.

1.1 Autômatos celulares

Autômatos celulares foram desenvolvidos inicialmente por John Von Neumann e Stanislaw Ulam, na década de 40, estando ligados a estudos sobre computação, inteligência e vida artificiais (Torrens, 2000). Podem ser considerados como um espaço finito composto por células organizadas em um grid, que mudam de estado automaticamente seguindo certas regras de transição, em função dos estados das células vizinhas. A interação desses componentes simples (células) com sua vizinhança pode gerar padrões de comportamento complexo, assim como ocorre com as cidades.
As possibilidades de espacialização dos autômatos celulares, aliadas a sua capacidade de representar processos dinâmicos, têm sido utilizadas como auxiliares na resolução de problemas ambientais e urbanos, como é o caso do crescimento espacial.

1.2 Grafos

A teoria de grafos provém da topologia (Sánchez, 1998), a qual se dedica a estudar relações entre pontos, linhas e superfícies, a partir de suas conexões. Grafos são conjuntos finitos de nodos conectados por arestas (Mariani, 2001) e são úteis para representar as relações de interação ou tensão espacial entre as células. Para isso, é necessário converter a estrutura do grid em um grafo, o que é feito considerando cada célula como um vértice e a vizinhança celular do tipo Moore (8 vizinhos) como as arestas de um grafo. Esta representação esta ilustrada na Figura 1.

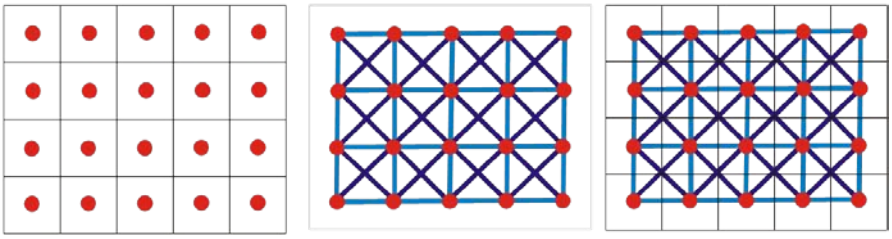

Fig. 1: a) representação celular; b) representação em grafos; c) células + grafos.

Enquanto autômatos celulares tratam de relações em nível de vizinhança imediata, grafos permitem representar estruturas espaciais a partir de vizinhanças remotas. Deste modo, a utilização de recursos de grafos com recursos de autômatos celulares permite tratar integradamente relações locais e relações globais.

1.3 SIG - Sistemas de Informações Geográficas

As informações disponíveis para a modelagem urbana são armazenadas em um ambiente SIG, responsável pela leitura, organização e visualização dos dados. Softwares de SIG permitem a execução de operações de elevada complexidade envolvendo dados espaciais

com relativa facilidade, aliada à possibilidade de desenvolvimento de novas ferramentas geoespaciais, através de geocomputação, destinadas à solução de diversos problemas específicos.
No SACI®, a integração com SIG é feita utilizando o software ArcView® 3.3, desenvolvido pela ESRI, ao qual o modelo de crescimento é instalado na forma de uma extensão. Assim, toda a entrada e saída de dados, bem como a parametrização de processos, é feita no ambiente do ArcView®. Essa integração foi possibilitada pela utilização da linguagem de programação própria do ArcView®, chamada Avenue®, que é uma linguagem do tipo script e permite acesso a um framework orientado a objetos de SIG.

1.4 O Simulador de Crescimento Urbano

O SACI® está implementado na forma de dois elementos conectados: uma biblioteca de ligação dinâmica (DLL – Dynamic Link Library), que contém o núcleo de processamento, e uma interface em ambiente SIG (Figura 2).

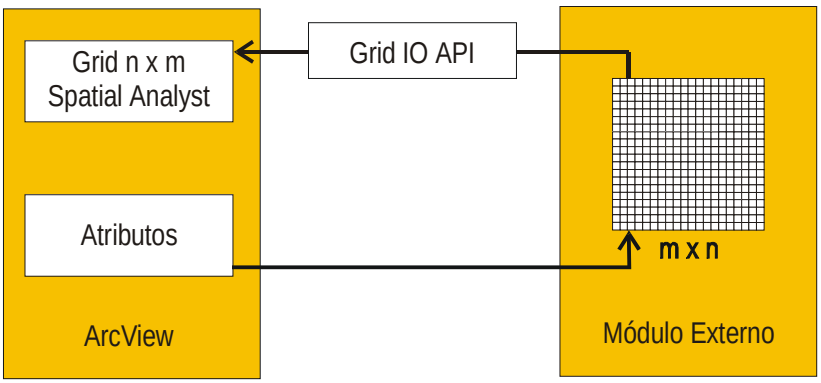

Fig. 2: integração entre o módulo externo do SACI® e o ArcView®.

O ArcView® é responsável pela entrada de dados referentes aos atributos e sua associação às células, armazenando estas informações em tabelas de banco de dados. A transferência dos dados do SIG para o módulo externo é feita através da biblioteca de funções Grid I/O, fornecida pela extensão Spatial Analyst da ESRI. Assim, o módulo externo acessa os grids e suas tabelas de atributos, transpondo esses dados a uma matriz sobre a qual ocorrerá o processamento. Posteriormente, o resultado do processamento é enviado de volta ao SIG para representação.

1.5 Situação atual

Após cinco anos do seu lançamento, é possível fazer uma análise da situação atual do SACI® e projetar novos rumos para o sistema. Um dos principais problemas do software atualmente é a lentidão do seu processamento. Outra questão a ser superada é a dependência de uma plataforma proprietária, no caso o ArcView®, o que dificulta a distribuição do sistema. O objetivo deste trabalho trata da superação destas limitações, bem como da implementação de novas funcionalidades ao sistema, atividades que não foram executadas devido à descontinuidade no desenvolvimento do software.

3 O CITYCELL®

A nova versão do SACI®, atualmente em desenvolvimento, está denominada CityCell®. A primeira etapa de desenvolvimento consiste na análise do algoritmo geral do sistema, de modo a encontrar as rotinas responsáveis pela maior parte do tempo de processamento demandado pelas simulações. Em uma primeira análise, foram encontrados três pontos críticos: a ausência de processamento paralelo, a interface entre o ArcView® e o SACI® e a implementação do algoritmo de busca de caminhos mínimos A*.

As questões de performance foram enfrentadas utilizando recursos de computação paralela e distribuída, otimização de algoritmos internos, técnicas avançadas de busca e armazenamento de caminhos mínimos e otimização do software para utilizar os novos recursos de hardware disponíveis nos computadores atuais, como os processadores de múltiplos núcleos.

Uma importante característica dos autômatos celulares é a de que se tratam de processadores paralelos, e não seriais, o que permite que o processamento de várias células seja feito simultaneamente. A primeira versão do sistema não utilizava esta característica, comportando-se como um processador serial. A implementação de processamento paralelo pode ser feita, neste caso, em dois níveis: 1) a execução de vários processos simultaneamente em um mesmo computador, levando em conta o número de núcleos disponíveis no processador; 2) a distribuição do processamento através de um cluster de computadores. No segundo caso, o grid pode ser dividido em pedaços menores e distribuído entre vários computadores via rede, que executam o processamento e devolvem os resultados para um servidor, que se encarregaria de consolidar os dados recebidos e os redistribuir aos computadores clientes a cada iteração.

A integração com o software ArcView® também gera uma perda de desempenho considerável no processamento. Por tratar-se de um software relativamente antigo, o ArcView® tem problemas de compatibilidade com sistemas operacionais e processadores mais novos, como as versões de 64 bits do Microsoft Windows®. Por outro lado, a falta de uma equipe permanente de desenvolvimento para o SACI® impossibilita que ocorra uma atualização constante, acompanhando os últimos lançamentos da ESRI, como o ArcGIS® 9.2 e 9.3. Esta limitação está superada com o lançamento de um sistema independente de plataformas proprietárias, facilitando a manutenção futura do software e ganhando em desempenho a medida que os processadores, sistemas operacionais e compiladores evoluem.

O CityCell está implementado com a utilização do ambiente de desenvolvimento CodeGear Delphi®, escolhida por permitir a criação de softwares de alta performance para o ambiente Windows®. Além disso, o Delphi® traz os recursos da VCL (Visual Component Library), um framework baseado em componentes que permite a criação de avançadas interfaces de usuário com facilidade e rapidez. A VCL é plenamente expansível, permitindo a instalação de componentes de terceiros bem como a criação de novos componentes.
A seguir, na Figura 3, é apresentada a estrutura do CityCell®:

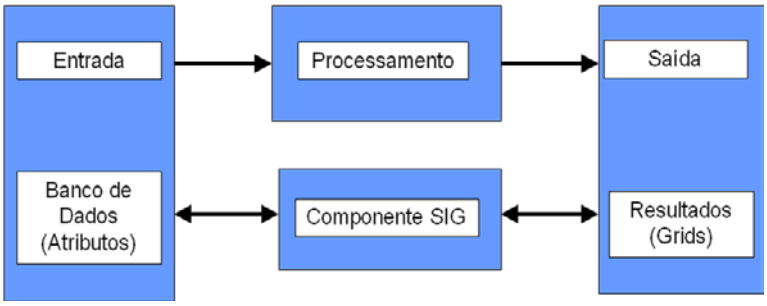

Fig. 3: estrutura do CityCell®.

A principal inovação desta versão do sistema é a substituição do ArcView® por um pacote de componentes de SIG próprio, integrado ao software. Este módulo é responsável por toda a interpretação, edição e visualização de dados raster necessários às simulações de crescimento urbano. Este conjunto de componentes foi desenvolvido como parte da VCL, portanto pode ser distribuído e utilizado em outros softwares que utilizem a linguagem Delphi®. A estrutura do pacote de componentes SIG é a seguinte (Figura 4):

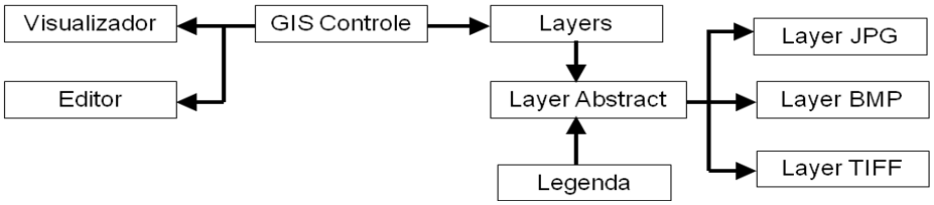

Fig. 4: estrutura dos componentes de SIG do CityCell®.

Os componentes visíveis para o usuário final são o Visualizador, que permite a visualização dos dados geoespaciais em formato raster, e o Editor, que permite a edição de grids célula por célula, permitindo uma maior flexibilidade na entrada de dados pelo usuário. O componente de Controle é responsável por fazer a ligação entre as layers e os componentes de visualização e edição. As layers são responsáveis por armazenar e interpretar os dados em diferentes formatos de arquivo, como JPEG, BMP e GeoTIFF. Suporte a novos formatos pode ser incluído futuramente com relativa facilidade, através da implementação de novos tipos de layers. Outra forma de incluir suporte a diferentes formatos de arquivos é através da conversão de grids em formatos menos populares para formatos padrão como GeoTIFF. Esta conversão pode ser feita manualmente pelo usuário, através do software de SIG em que o grid original foi gerado, ou automaticamente pelo SACI® 2.0 através da integração com a biblioteca de código aberto GDAL - Geospatial Data Abstraction Library (OSGF, 2009), que permite a tradução de dados geoespaciais entre diversos formatos raster.

3.1 Busca de caminhos mínimos

No modelo de crescimento, em cada iteração o sistema calcula as tensões espaciais e o caminho mínimo entre todos os pares possíveis de células. O caminho mínimo entre duas células é calculado utilizando o algoritmo de busca heurística A*, levando em conta atrações e resistências existentes nas células do grid. A quantidade de vezes em que esta busca é executada, aliada à complexidade do algoritmo, torna este um ponto crítico no desempenho geral do sistema.

De modo a minimizar o impacto da busca de caminhos no desempenho da simulação, estão implementadas as seguintes técnicas para ganho de velocidade, descritas por Lester (2003): a) utilização de vetores unidimensionais para armazenamento de listas, ao invés de estruturas de dados dinâmicas e orientadas a objetos, que podem consumir uma parcela de tempo maior necessária para a criação e manutenção de tais objetos; b) manutenção da lista aberta utilizando uma binary heap, o que aumenta o desempenho exponencialmente nos casos de grids maiores.
Posteriormente a essas otimizações, foram buscadas novas formas de diminuir o tempo de execução da busca de caminhos mínimos. A primeira técnica utilizada neste caso, com ganhos significativos de performance, foi o armazenamento de caminhos já calculados. Neste caso o sistema inicia a busca de caminhos entre células mais distantes espacialmente, ao invés de começar pelas células mais próximas. Uma vez encontrado o caminho mínimo entre estas células, o caminho entre a célula de origem e todas as células intermediárias estará automaticamente calculado, bastando ser armazenado em memória para ser utilizado novamente quando necessário. Este processo está ilustrado na Figura 5, adiante.

Fig. 5: a – célula de origem; b – célula de destino; pintadas em cinza, o caminho entre a e b, que deve ser armazenado em memória para uso posterior.

A implementação do reaproveitamento de caminhos mínimos trouxe ganhos significativos de desempenho para o sistema, diminuindo o tempo gasto com essa etapa de processamento em aproximadamente 50%. A execução de várias buscas em paralelo melhorou ainda mais esse índice, chegando a ganhos de desempenho da ordem de 85%, como pode ser observado na Tabela 1.

Tabela 1 Comparação entre os tempos de execução do algoritmo A*, considerando a memorização de caminhos já pesquisados e a utilização de processamento paralelo

	Tempo			
Tamanho do GRID	**A* Normal**	**A* Memo**	**A* Memo Paralelo**	**Ganho**
32 x 32	13,906s	6,938s	2,351s	83%
48 x 48	1min58,487s	54,177s	17,971s	85%
64 x 64	8min51,044s	3min52,452s	1min22,027s	85%
96 x 96	1h15min08,855s	32min38,419s	11min28,282s	85%

Como forma de melhorar ainda mais a performance da busca de caminhos mínimos, foi considerada a substituição do algoritmo implementado. O A* é o algoritmo mais utilizado em casos em que a origem e o destino da busca são conhecidos e não seguem um padrão pré-definido de localização, pois o mesmo utiliza heurística para diminuir o número de células visitadas durante uma execução (Lester, 2003). No caso específico do CityCell, as sucessivas buscas efetuadas seguem um padrão previsível, abrangendo todas as células que possuem algum tipo de carregamento. Neste caso, a utilização do algoritmo de Dijkstra (Corben et al, 2001) se mostra mais favorável, permitindo que uma única busca retorne todos os caminhos de uma determinada célula para todas as outras do grid, armazenando essas informações em uma matriz. Assim, basta executar o algoritmo uma única vez para cada célula do grid e simplesmente percorrer a matriz armazenada em memória para encontrar o caminho da célula de origem para qualquer outra do grid. Esta técnica diminuiu drasticamente o tempo gasto com buscas de caminhos mínimos, com ganhos de mais de 99% em relação ao algoritmo utilizado na primeira versão do SACI, conforme Tabela 2, adiante:

Tabela 2 Comparação entre os tempos de execução do algoritmo A* com o algoritmo Dijkstra.

	Tempo			
Tamanho do GRID	**A* Normal**	**Dijkstra**	**Dijkstra Paralelo**	**Ganho**
32 x 32	13,906 s	0,546s	0,218s	98%
48 x 48	1min58,487s	2,933s	1,217s	99%
64 x 64	8min51,044s	9,672s	4,275s	99%
96 x 96	1h15min08,855s	56,722s	28,392s	99%

4 CONSIDERAÇÕES

Analisando o modelo de crescimento SACI® foi possível entender seu funcionamento e identificar a possibilidade de ganhos de performance significativos para a segunda versão do sistema, denominada CityCell®. As questões de performance foram enfrentadas utilizando recursos de computação paralela e distribuída, otimização de algoritmos internos, técnicas avançadas de busca e armazenamento de caminhos mínimos e otimização do software para utilizar os novos recursos de hardware disponíveis nos computadores atuais, como os processadores de múltiplos núcleos.

Já a questão da dependência do software ArcView® foi eliminada no CityCell, que funciona de forma independente e implementa internamente as funcionalidades de entrada, saída e parametrização de processos, que antes eram feitas através de recursos disponíveis no ArcView®. O CityCell inclui compatibilidade com os formatos de arquivo mais comuns em ambiente SIG, o que facilita o aprendizado e a utilização do modelo por parte dos usuários, elimina a necessidade de aquisição de um software proprietário de custo elevado e resolve questões de manutenção e compatibilidade do sistema com computadores e sistemas operacionais mais novos. Essas atualizações tem por objetivo trazer um significativo ganho de produtividade aos pesquisadores usuários do CityCell, de modo a aumentar sua popularidade na comunidade científica.

5 REFERÊNCIAS

CORMEN, T. H.; LEISERSON, C. E.; RIVEST R. L.; STEIN, C. (2001) **Introduction To Algorithms, 2nd Edition.** MIT Press.

KRAFTA, R. (1994). **Modelling Intraurban configurational development.** Environment and Planning B: Planning and Design, v. 21. London: Pion. p. 67-82.

LESTER, P. (2003). **Using Binary Heaps in A* Pathfinding.** [disponível em 24 de julho de 2008 em www.policyalmanac.org/games/binaryHeaps.htm]

MARIANI, A. C. (2001). **Teoria dos Grafos.** UFSC/CTC/INE. [disponível em 20 de agosto de 2008 em www.inf.ufsc.br/ grafos/livro.html]

OSGF – Open Source Geospatial Foundation. **GDAL - Geospatial Data Abstraction Library.** [disponível em 05 de maio de 2009 em http://www.gdal.org]

POLIDORI, M. C. (2004). **Crescimento urbano e ambiente: um estudo exploratório sobre as transformações e o futuro da cidade.** UFRGS, PPGECO.

TORRENS, P. (2000). **How cellular models of urban systems work.** London: Casa, U-CL. 75 p. [disponível em 20 de agosto de 2008 em http://www.casa.ucl.ac.uk/working_papers.htm]