Relatório da Disciplina MAC 5701 - Tópicos em Ciência da Computação

Marco Aurélio Stefanes
Orientador: José Augusto Ramos Soares

13 de Julho de 2000

No decorrer do semestre selecionamos e discutimos alguns dos artigos abaixo. Segue a bibliografia anotada conforme descrita no plano inicial de estudo da disciplina. Este trabalho cumpriu seu objetivo, a saber, propiciar uma visão mais detalhada do resultados recentes na nossa área de pesquisa.

Artigo: Andrews, M. G.; Atallah, M. J.; Chen, D. Z.; Lee, D. T. "Parallel algorithms for maximum matching in complements of interval graphs and related problems," Algorithmica 26 (2000), no. 2, 263–289.

Resumo: Dado um conjunto de n intervalos representando um grafo de intervalos, o problema de encontrar um emparelhamento máximo entre pares de intervalos disjuntos tem sido tratado no modelo seqüencial. Neste artigo os autores apresentam algoritmos paralelos para computar emparelhamentos de cardinalidade máxima entre pares de intervalos disjuntos em grafos de intervalos nos modelos EREW PRAM e hipercubo. Para o caso geral, os algoritmos computam um emparelhamento máximo em tempo $O(\log^3 n)$ usando $O(n/\log^2 n)$ processadores no EREW PRAM e usando n processadores no hipercubo. Para o caso dos grafos de intervalos próprios, os algoritmos rodam no EREW PRAM em tempo $O(\log n)$ usando $O(n)$ processadores se os intervalos de entrada não estão ordenados e usando $O(n/\log n)$ processadores caso contrário. Em um hipercubo de n processadores, o algoritmo para o caso próprio toma tempo $O(\log n \log\log n)$ para entrada não-ordenada e tempo $O(\log n)$ para entrada ordenada.

Artigo: Bampis, E.; Giannakos, A.; Karzanov, A.; Manoussakis, Y.; Milis, I. "A parallel algorithm for finding a perfect matching in a planar graph," Parallel Process. Lett. 8 (1998), no. 3, 399–405.

Resumo: Os autores descrevem um algoritmo CRCW PRAM simples e intuitivo para encontrar um emparelhamento perfeito em um grafo planar que toma tempo $O(\log^2 n \log \Phi)$ e $O(n^2 M(n))$ processadores, onde n é a ordem, Φ é o número de emparelhamentos perfeitos disjuntos do grafo e $M(n)$ é a complexidade do melhor algoritmo seqüencial para multiplicação de matrizes de ordem n.

Artigo: Bera, Debashis; Pal, Madhumangal; Pal, Tapan K. "An optimal parallel algorithm to compute all cutvertices and blocks on permutation graphs," Int. J. Comput. Math. 72 (1999), no. 4, 449–462.

Resumo: Neste artigo, um conjunto de intervalos é construído através da projeção da representação geométrica de um grafo de permutação para encontrar todos os vértices de cortes e componentes. Algoritmos seqüenciais e paralelos são apresentados para encontrar todos os vértices de corte e componentes destes grafos. O algoritmo seqüencial toma tempo $O(n)$ e o algoritmo paralelo toma tempo $O(\log n)$ usando $O(n/\log n)$ processadores no modelo EREW PRAM, onde n representa o número de vértices do grafo.

Artigo: Bose, P.; Chan, A.; Dehne, F. and Latzel, M. " Coarse grained parallel maximum matching in convex bipartite graphs", in Proc. 13th International Parallel Processing Symposium (IPPS'99), 1999, pp. 125-129.

Resumo: Os autores apresentam um algoritmo CGM paralelo para computar um emparelhamento máximo de um grafo bipartido convexo $G = (A, B, E)$. Para p processadores com N/p memória por processador, $N = |A|+|B|, N/p >= p$, o algoritmo requer $O(\log p)$ rodadas de comunicação e computação local $O(T_{seq}(n/p, m/p)+n/p \log p)$, onde $n = |A|, m = |B|$ e $T_{seq}(n, m)$ é a complexidade seqüencial para o problema. Para o modelo BSP isto implica $O(\log p)$ superpassos com custo de comunicação $O(gN + gn/p \log p)$ e computação local $O(T_{seq}(n/p, m/p)+n/p \log p)$.

Artigo: Caceres, E.; Chan, A.; Dehne, F.; and Prencipe, G. "Coarse Grained Parallel Algorithms for Detecting Convex Bipartite Graphs", IPPS'97

Resumo: Os autores apresentam algoritmos para os modelos CGM e BSP para resolver dois problemas em grafos: (1) Determinar se um grafo é bipartido e (2) determinar se um grafo bipartido é convexo. Os algoritmos tomam $O(\log p)$ e $O(\log^2 p)$ rodadas respectivamente, e trabalho seqüencial linear por rodada no CGM com p processadores e $N = |G|$. Os algoritmos assumem que $N/p \geq p^\epsilon$ para algum $\epsilon > 0$ fixo. Os resultados implicam em algoritmos BSP com $O(\log p)$ e $O(\log^2 p)$ superpassos, respectivamente, $O(g \log(p)N/p)$ tempo de comunicação, e $O(\log(p)N/p)$ tempo de computação local. O algoritmo para determinar se um grafo bipartido é convexo inclui uma nova versão CGM da estrutura de dados *PQ tree.* Assim o algoritmo também resolve, com a mesma complexidade de tempo indicada acima, o problema de testar a propriedade uns consecutivos para matrizes (0,1), bem como para o reconhecimento de grafos cordais.

Artigo: Caceres, E.; Dehne F.; Ferreira, A.; Flocchini, P.; Rieping, I.; Roncato, A.; Santoro, N.; and Song, S. W. " Efficient parallel graph algorithms for coarse grained multicomputers and BSP" , ICALP'97, LNCS 1256.

Resumo: Neste artigo, os autores apresentam algoritmos CGM e BSP para os seguintes problemas: (1) list raking, (2) construção do Passeio de Euler, (3) computação de componentes conexas e floresta geradora, (4) preprocessamento do menor ancestral comum, (5) contração de árvores e avaliação de árvore de expressão (6) computação de uma decomposição em orelhas, (7) 2-aresta conexidade e biconexidade e (8) reconhecimento de grafos cordais. Os algoritmos para os problemas 1-7 gastam $O(\log p)$ rodadas de comunicação e trabalho linear por rodada, já o problema 8 gasta $O(\log n \log p)$ rodadas de comunicação e trabalho também linear por rodada.

Artigo: Caceres, Edson; Chan, Albert; Dehne, Frank; Song, Siang W. "Coarse Grained Parallel

Graph Planarity Testing", Preliminary Version.

Resumo: Neste artigo é apresentado um algoritmo paralelo de granularidade grossa para testar planaridade e imersão planar. O algoritmo requer $O(log^2 p)$ rodadas de comunicação e trabalho seqüencial linear por rodada. Ele assume que a memória local por processador, N/p, é maior que p^ϵ para algum $\epsilon > 0$ fixo. O algoritmo está baseado em uma estrutura de dados de um método PRAM apresentado por Klein, usando uma implementação paralela de PQ-trees. A principal contribuição deste artigo é o estudo de vários passos PRAM que são ineficientes nas máquinas paralelas existentes. Neste artigo é apresentado implementações CGM não-triviais e eficientes de várias partes da estratégia geral proposta por Klein. O principal resultado é um algoritmo paralelo de teste de planaridade mais eficiente nas máquinas existentes

Artigo: Chan, A.; Dehne, F.; and Rau-Chaplin, A. " Coarse grained parallel geometric search", Journal of Parallel and Distributed Computing, Vol. 57, No. 2, 1999, pp. 224-236.

Resumo: Os autores descrevem um algoritmo paralelo para resolver o *problema da busca do próximo elemento* em um conjunto de segmentos de retas, usando o modelo CGM. O algoritmo requer $O(1)$ rodadas de comunicação e $O((n/p)\log p)$ memória por processador, assumindo $n/p \geq p$. Os resultados implicam em soluções para a localização de pontos, decomposição trapezoidal e triangularização de polígonos. Uma versão simplificada para segmentos paralelos em x requer apenas $O(n/p)$ memória por processador, e uma implementação desta versão é discutida. Como no artigo de Devillers e Fabri, este algoritmo é baseado em uma implementação distribuída de árvores de segmentos de tamanho $O(n \log n)$. O algoritmo apresentado melhora o algoritmo de Devillers e Fabri nos seguintes aspectos: (1) ele é mais geral, que resolve, por exemplo o localização de pontos no plano. (2) o algoritmo toma $O((n/p)\log n)$ computação local, ao invés de $O(\log p * (n/p)\log n)$. (3) o algoritmo requer $O((n/p)\log p)$ memória local, ao invés de $O((n/p)\log n)$.

Artigo: Chandran, Sharat; Kim, Sung Kwon; Mount, David M. "Parallel computational geometry of rectangles," Algorithmica 7 (1992), no. 1, 25–49.

Resumo: Este artigo estuda vários algoritmos PRAM para retângulos *upright* (com lados paralelos na coordenada x) no plano que são motivados principalmente pelas aplicações em processamento de imagens. Especificamente, os seguintes problemas para um conjunto Q de retângulos *upright* no plano são resolvidos de forma ótima em tempo $O(\log n)$ usando n processadores, e em espaço $O(n \log n)$: Computar a área, o perímetro e os momentos de inércia de $\bigcup Q$; Computa o clique máximo de Q; computa as componentes conexas de $\bigcup Q$. Além disso, são dados algoritmos de tempo $O(n)$ para determinar o *contour* de Q e a representação x medial de uma imagem binária $n \times n$. Os algoritmos são baseados em uma estrutura de dados chamada árvore plane-sweep ortogonal, a qual, pode ser construída em tempo $\Theta(\log n)$.

Artigo: Czumaj, Artur; Diks, Krzysztof; Przytycka, Teresa M. "Parallel maximum independent set in convex bipartite graphs," Inform. Process. Lett. 59 (1996), no. 6, 289–294.

Resumo: Um grafo bipartido $G = (V, W, E)$ é chamado convexo se os vértices em W podem ser ordenados de tal forma que os elementos de W adjacentes a qualquer vértice em $v \in V$ formam um intervalo (i.e. uma seqüência de vértices numerados consecutivamente). Tal grafo pode ser representado em uma forma compacta que requer espaço $O(n)$, onde $n = \max\{|V|, |W|\}$.

Dado um grafo bipartido convexo G na forma compacta, Dekel e Sahni descreveram um algoritmo EREW PRAM de tempo $O(\log^2(n))$ usando n processadores para encontrar um emparelhamento máximo em G. Este artigo mostra que este emparelhamento pode ser usado para construir de forma ótima um conjunto independente máximo. O algoritmo deste artigo roda em tempo $O(\log n)$ com $n/\log n$ processadores em um CRCW PRAM arbitrário.

Artigo: Dehne F. and Goetz, S. "Practical parallel algorithms for minimum spanning trees", in Proc. 17th IEEE Symp. on Reliable Distributed Systems, Advances in Parallel and Distributed Systems, 1998, pp. 366-371.

Resumo: Os autores estudam algoritmos para computar a árvore geradora mínima de um grafo G de n vértices , m arestas e com peso nas arestas. Os autores consideram um grafo de entrada com $m/n > p$, onde p é o número de processadores. Para este caso, eles mostram que algoritmos simples com padrões de comunicação independente dos dados são eficientes, tanto na teoria quanto na prática. Os algoritmos são avaliados teoricamente usando o modelo BSP e empiricamente através de implementações.

Artigo: Dehne, F. and Guimaraes, K. "Exact and approximate computational geometry solutions of an unrestricted point set stereo matching problem" , Information Processing Letters 64, 1997, pp. 107-114.

Resumo: Neste artigo os autores estudam o problema de computar uma solução exata e arbitrariamente próxima da exata do problema de emparelhamento stereo de um conjunto de pontos irrestritos. No contexto da abordagem clássica como o algoritmo de Marr-Poggio, isto significa que o artigo estuda como resolver os subproblemas básicos irrestritos criados com tais abordagens, possivelmente produzindo uma performance geral melhorada destes métodos. Os autores apresentam um algoritmo que toma tempo $O(n^{2+4k})$ e espaço $O(n^4)$ para o emparelhamento stereo irrestrito exato, onde n representa o número de pontos em cada conjunto e k o número de níveis de profundidade considerados. Os autores generalizam a noção de uma solução δ-aproximada para congruência de um conjunto de pontos para o problema de emparelhamento stereo e apresentam um algoritmo δ-aproximado que gasta tempo $O((\epsilon/\delta)^k n^{2+2k})$ e espaço $O((\epsilon/\delta)n^2)$ para o emparelhamento stereo irrestrito (ϵ representa a medida de imprecisão da imagem). Os Autores introduzem para isto algumas novas ferramentas de geometria computacional para o emparelhamento stereo.

Artigo: Dehne, F. and Klein, R. "The big sweep: On the power of the beach line approach for Voronoi diagrams", Algorithmica, Vol. 17, 1997, pp. 19-32.

Resumo: Os autores mostram que a abordagem *wavefront* para diagramas de Voronói (um algoritmo determinístico que não usa transformação geométrica) pode ser generalizado para medidas de distância mais gerais que a métrica euclidiana. De fato, os autores fornecem o primeiro algoritmo ótimo no pior caso (tempo $O(n \log n)$ e espaço $O(n)$) que é válido para as classes *nice metrics* no plano. *Nice metrics* incluem todas as funções de distância convexa além da métrica Moscow e métricas compostas. O algoritmo é conceitualmente simples, mas trata de todas as possíveis deformações do diagrama.

Artigo: Dehne, F.; Fabri, A. and Rau-Chaplin, A. "Scalable parallel computational geometry for coarse grained multicomputers", International Journal on Computational Geometry, Vol. 6, No. 3, 1996, pp. 379 - 400.

Resumo: Os autores estudam algoritmos CGM. Eles apresentam algoritmos paralelos de tempo $O(T_{seq}/p + T_s(n,p))$ para vários problemas geométricos. $T_s(n,p)$ é o tempo de uma operação de ordenação global. Os resultados são independentes da rede de interconexão. Estes métodos possuem vantagens práticas importantes: Para comunicação interprocessadores eles usam um número fixo de operações de roteamentos globais e todas as demais operações são computadas localmente. Além disso, os algoritmos usam um pequeno número de grandes mensagens, o qual reduz o overhead de comunicação. As implementações mostram que estes métodos são fáceis de implementar e dão bons resultados de tempo.

Artigo: Fabri, Andreas; Devillers, Olivier "Scalable algorithms for bichromatic line segment intersection problems on coarse grained multicomputers". Geometric constraints: theory and practice (1993). Internat. J. Comput. Geom. Appl. 6 (1996), no. 4, 487–506.

Resumo: Os autores apresentam algoritmos paralelos escaláveis para os problemas de interseção de segmentos de linhas bicromáticas para o modelo CGM. Considerando $n \geq p^2$, onde n é o número de segmentos de linhas e p, o número de processadores, os autores obtêm um algoritmo de contagem de inteseções de complexidade $O((n \log n \log p)/p + T_s(n \log p, p))$, onde $T_s(m,p)$ é o tempo usado para ordenar m itens com p processadores. Além disso, é gasto um tempo adicional seqüencial de $O(k/p)$ para relatar as k interseções. Como o algoritmo seqüencial gasta tempo $O(n \log n)$ para contar e $O(k)$ para relatar, o speedup obtido é de $p/\log p$.

Artigo: Ghouse, Mujtaba R.; Goodrich, Michael T. "Fast randomized parallel methods for planar convex hull construction," Comput. Geom. 7 (1997), no. 4, 219–235.

Resumo: Os autores apresentam um número de algoritmos paralelos eficientes para construir fecho convexo 2D em um CRCW PRAM probabilístico. Especificamente, os autores mostram como construir um fecho convexo de n pontos ordenados no plano em tempo $O(1)$ usando trabalho $O(n \log n)$ com probabilidade n-exponencial, ou alternativamente, em tempo $O(\log^* n)$ usando trabalho $O(n)$ com probabilidade n-exponencial. É mostrado também, como encontrar um fecho convexo de n pontos não ordenados no planos em tempo $O(\log n)$ usando trabalho $O(n \log h)$ com probabilidade n-exponencial, onde h é o número de arestas no fecho convexo. Os algoritmos para entradas não ordenadas dependem do uso de novos procedimentos "in-place", isto é, procedimentos que são definidos em um subconjunto de elementos de entrada e que trabalham sem ter de reordenar a entrada. Para atingir os limites n-exponenciais, os autores usam uma nova técnica chamada "failure sweeping".

Artigo: Goodrich, M. T.; Ghouse, M. R.; Bright, J. "Sweep methods for parallel computational geometry", Special issue on parallel computing (Crete, 1990). Algorithmica 15 (1996), no. 2, 126–153.

Resumo: Os autores usam plane sweeping paralelo com o intuito de fornecer algoritmos paralelos eficientes para um número de problemas em geometria computacional. A abordagem é ilustrada com um número de aplicações que incluem: eliminação de superfícies escondidas, avaliação de fronteira CSG, computação do contorno de uma coleção de retângulos, e eliminação de superfícies escondidas para retângulos.

Artigo: Goodrich, Michael T. "Randomized fully-scalable BSP techniques for multi-searching and convex hull construction" (preliminary version). Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (New Orleans, LA, 1997), 767–776, ACM, New York, 1997.

Resumo: O autor estuda técnicas para desenvolver algoritmos eficientes para o modelo BSP. Ele apresenta técnicas que produzem eficiência ótima para todos os valores de p. Em particular, são tratados dois problemas: multibusca e construção do fecho convexo. Os métodos aqui apresentados resultam em algoritmos de tempo $O((n \log n)/p)$ e, para $h = \Theta(n/p)$, o número de rodadas de comunicação é $O((\log n)/\log(h+1))$, com alta probabilidade.

Artigo: Honma, H. "An optimal parallel algorithm for constructing a spanning forest on interval graphs", Res. Rep. Kushiro Natl. Coll. Technol. No. 32, (1998), 55–58.

Resumo: Seja $G = (V, E)$ um grafo simples com n nós, m arestas e p componentes conexas. O problema de construir uma floresta geradora é encontrar uma árvore geradora para cada componente conexa de G. Para um grafo simples, Chin *et al* [Comm. ACM 25 (1982), no. 9, 659–665] demonstraram que uma floresta geradora pode ser encontrada em tempo $O(\log^2 n)$ usando $O(n^2/\log^2 n)$ processadores. Neste artigo, os autores propõem um algoritmo paralelo de tempo $O(\log n)$ com $O(n/\log n)$ processadores no EREW PRAM para construir uma floresta geradora em grafos de intervalos.

Artigo: Karypis, George; Kumar, Vipin "Parallel multilevel k-way partitioning scheme for irregular graphs", SIAM Rev. 41 (1999), no. 2, 278–300.

Resumo: Os autores apresentam uma formulação paralela para um algoritmo de particionamento de grafos k-way em multinível. A característica chave desta formulação paralela é que ela é capaz de produzir um alto grau de concorrência enquanto mantém a alta qualidade das partições produzidas pelo algoritmo de particionamento k-way em multinível serial. Em particular, o tempo tomado pelo algoritmo de particionamento de grafo paralelo é ligeiramente maior que o tempo tomado para redistribuir o grafo entre os processadores de acordo com a nova partição. Experiências com uma variedades de grafos de elementos finitos mostram que esta formulação paralela produz particionamento de alta qualidade em um curto espaço de tempo. Por exemplo, um particionamento 128-way de grafos com um milhão de vértices pode ser computado em pouco mais de 2 segundos no 128-processor Cray T3D. Além disso, a qualidade das partições produzidas é comparável àquelas produzidas pelo algoritmo k-way em multinível serial. Assim, estes algoritmos paralelos tornam viáveis reparticionamentos freqüentes de grafos em computações dinâmicas sem comprometer a qualidade do particionamento.

Artigo: Kim, Hiryoung; Sprague, Alan P. "Optimal parallel matching on bipartite permutation graphs," Parallel Algorithms Appl. 10 (1997), no. 3-4, 165–176.

Resumo: Os autores apresentam um algoritmo paralelo de custo ótimo para o problema do emparelhamento máximo em grafos de permutações bipartidos no EREW PRAM. Chen and Yesha resolveram o emparelhamento para grafos bipartidos convexos em tempo $O(\log^2 n)$ usando $O(n)$ processadores. Dado um diagrama de permutação, o algoritmo do artigo roda em tempo $O(\log n)$ usando $O(n/\log n)$ processadores. Este método começa com um algoritmo guloso facilmente compreendido. Os Autores definem uma operação binária não-trivial a qual é associativa e equivalente ao algoritmo guloso. Assim, uma soma prefixa paralela pode ser aplicada ao problema.

Artigo: Ma, Jun; Ma, Shaohan; Iwama, Kazuo; Gu, Qianping "A parallel algorithm for generating multiple ordering spanning trees in undirected weighted graphs", Acta Math. Appl. Sinica (English Ser.) 15 (1999), no. 3, 303–309.

Resumo: Neste artigo, os autores propõem um algoritmo paralelo eficiente para gerar $k(k \geq 1)$ árvores geradoras de um grafo conexo, com peso e não dirigido $G = (V, E, W)$ em ordem crescente de pesos. Ele roda em tempo $O(T_{\text{mst}}(n) + k \log n)$ com $O(n^2/\log n)$ processadores no CREW PRAM, onde $n = |V|, m = |E|$ e $T_{\text{mst}}(n)$, $O(\log n) \leq T_{\text{mst}}(n) \leq O(\log^2 n)$, é o tempo de execução do algoritmo paralelo mais rápido para encontrar uma MST de G no CREW PRAM.

Artigo: Nakayama, Shin-ichi; Masuyama, Shigeru "A parallel algorithm for finding a minimum-weight connected dominating set on trapezoid graphs," Math. Japon. 45 (1997), no. 1, 165–171.

Resumo: Um conjunto dominante D em um grafo $G = (V, E)$ é um conjunto de vértices em V tal que todo vértice em $V - D$ é adjacente a algum vértice em D. Um conjunto dominante é conexo se o grafo induzido por D é conexo. O Problema do conjunto dominante convexo de peso mínimo é o problema de encontrar um conjunto dominante convexo com o menor peso total de vértices em D. Este problema é sabido ser NP-Difícil para grafos em geral. Assim, restringindo a classe de grafos, alguns algoritmos seqüenciais polinomiais tem sido desenvolvidos. Neste artigo, os autores propõem um algoritmo paralelo eficiente para encontrar um conjunto dominante convexo de peso mínimo para grafos trapezoidais em tempo $O(\log^2 n)$ usando $O(n^3)$ processadores no CREW PRAM.

Artigo: Nakayama, Shin-ichi; Masuyama, Shigeru "A parallel algorithm for solving the coloring problem on trapezoid graphs," Inform. Process. Lett. 62 (1997), no. 6, 323–327.

Resumo: O problema da coloração é colorir um grafo de entrada com um número mínimo de cores tal que vértices adjacentes tenham cores diferentes. Este problema apesar de NP-difícil pode ser resolvido em tempo polinomial para certas classes de grafos: grafos de intervalos, de permutação e trapezoidais. Neste artigo, os autores propõem um algoritmo paralelo para resolver o problema de coloração em grafos trapezoidais G os quais rodam em tempo $O(\log^2 n)$ usando $O(n^3/\log n)$ processadores no CREW PRAM onde n é o número de vértices de G.

Artigo: Nakayama, Shin-ichi; Masuyama, Shigeru "A simple near optimal parallel algorithm for recognizing outerplanar graphs," J. Math. Tokushima Univ. 30 (1996), 71–80.

Resumo: Um grafo outerplanar é um grafo que pode ser imerso no plano tal que todos os vértices situam-se na fronteira da face exterior. Neste artigo, os autores descrevem um algoritmo paralelo simples e próximo do ótimo para reconhecer se um dado grafo G é outerplanar em tempo $O(\log n)$ usando $O(n\alpha(l, n)/\log n)$ processadores em um CRCW PRAM árbitrário para $l = O(n)$. Embora um algoritmo paralelo próximo do ótimo para grafos em geral possa também ser obtido pela combinação do algoritmo de [K. Diks, T. Hagerup and W. Rytter, in Mathematical Foundations of Computer Science 1989 (Porabka-Kozubnik, 1989), 207–217, Lecture Notes in Comput. Sci., 379, Springer, Berlin, 1989;] com um algoritmo para encontrar componentes biconexas [D. S. Fussell, V. Ramachandran and R. Thurimella, SIAM J. Comput. 22 (1993), no.

3, 587–616;] este método é completamente diferente daqueles e o algoritmo é muito mais simples.

Artigo: Pal, Madhumangal; Bhattacharjee, G. P. “An optimal parallel algorithm for all-pairs shortest paths on unweighted interval graphs,” Nordic J. Comput. 4 (1997), no. 4, 342–356.

Resumo: É apresentado um algoritmo paralelo ótimo para encontrar todos os pares de caminhos mínimos em um grafo de intervalos sem pesos que toma tempo $O(n/p + \log n)$ no EREW PRAM, onde p e n representam, respectivamente, o número de processadores e o número de vértices do grafo.

Artigo: Pal, Madhumangal; Bhattacharjee, G. P. “An optimal parallel algorithm to color an interval graph”, Parallel Process. Lett. 6 (1996), no. 4, 439–449

Resumo: É apresentado um algoritmo paralelo ótimo para colorir grafos de intervalos usando o número mínimo de cores. O algoritmo paralelo proposto toma tempo $O(n/p + \log n)$ com p processadores no EREW PRAM.

Artigo: Ramachandran, Vijaya “Parallel algorithms for reducible flow graphs”, J. Algorithms 23 (1997), no. 1, 1–31.

Resumo: Os autores apresentam algoritmos paralelos NC para reconhecimento de grafos de fluxo redutível (RFG) e para encontrar dominadores, conjunto de vértices de *feedback* mínimo e uma árvore de busca em profundidade em um RFG. Em uma RFG de n-nós, todos estes algoritmos rodam em tempo polilogarítmico usando $M(n)$ processadores, onde $M(n)$ é o número de processadores necessários para multiplicar duas matrizes $n \times n$ em tempo polilogarítmico. Os autores mostram que encontrar um conjunto de vértices de *feedback* mínimo em um RFG com peso nos vértices ou encontrar um um conjunto de arcos de *feedback* mínimo em RFG com peso nos arcos é P-completo. Para pesos nos arcos e vértices, os autores apresentam algoritmos RNC para estes problemas e mostram que os problemas estão em NC se, e somente se, o problema de encontrar o corte mínimo em uma rede de fluxo está em NC.

Artigo: Ramachandran, Vijaya; Reif, John “Planarity testing in parallel”, J. Comput. System Sci. 49 (1994), no. 3, 517–561.

Resumo: Os autores apresentam um algoritmo paralelo CRCW PRAM que ou constrói uma imersão planar de um dado grafo ou determina que o grafo não é planar. O algroritmo é baseado em uma decomposição em orelhas do grafo. O algoritmo roda em tempo $O(\log n)$ para um grafo de n nós. O número de processadores requerido é limitado por $O(n)$ e por: (1) o número de processadores necessários para realizar o bucketsort em n números de n bits em tempo $O(\log n)$, e (2) o número de processadores necessários para encontrar componentes conexas de um grafo com n nós em tempo $O(\log n)$. Se os passos são realizados seqüencialmente, o algoritmo roda em tempo $O(n)$. Os autores mostram que o algoritmo pode realizar trabalho $O(n)$ e ter seu tempo de processamento limitado superiormente pelo máximo entre $O(\log n)$, o tempo mais rápido para um algoritmo bucketsort de trabalho linear e o mais rápido algoritmo para componentes conexas de trabalho linear.

Artigo: Reif, John H.; Sen, Sandeep “Optimal randomized parallel algorithms for computational geometry”, Algorithmica 7 (1992), no. 1, 91–117.

Resumo: Os autores apresentam algoritmos paralelos para alguns problemas em geometria computacional os quais tem tempo de execução $O(\log n)$ usando n processadores, com alto grau de probabilidade. Entre os problemas incluem: localização de ponto no plano, triangularização e decomposição trapezoidal. Os autores também apresentam algoritmos ótimos para máximos tridimensinais e contagem de domínio de conjunto através da aplicação de ordenação inteira. Muitos destes algoritmos rodam no modelo CREW PRAM e tem um ótimo produto tempo-processador, que melhora o melhor algoritmo previamente conhecido. O cerne deste algoritmo é uma estrutura de dados que emula o paradigma plane-sweeping usado por algoritmos seqüenciais. Os autores estendem algumas das técnicas usadas por Reischuk [in 22nd Annual Symposium on Foundations of Computer Science (Nashville, TN, 1981), 212–219, IEEE, New York, 1981] para o algoritmo *flashsort* para realizar divisão e conquista no plano eficientemente, levando a melhora de performance do método.

Artigo: Reif, John H.; Sen, Sandeep "Randomized algorithms for binary search and load balancing on fixed connection networks with geometric applications", SIAM J. Comput. 23 (1994), no. 3, 633–651.

Resumo: Atualmente há um número de problemas em geometria computacional que possui um algoritmo ótimo no modelo PRAM. Este artigo apresenta algoritmos paralelos probabilísticos que rodam em uma rede *butterfly* de n processadores em tempo $O(\log n)$ para os seguintes problemas de entrada de tamanho n: decomposição trapezoidal, visibilidade, triangularização e fecho convexo 2D. Exceto o algoritmo de fecho convexo 2D, os demais são os primeiros algoritmos geométricos não triviais que obtem esta performance em rede de interconexão fixa. Estas técnicas usam um número de idéias de *flashsort* que foram modificadas para manipular situações mais difíceis.