Aplicação de Arquiteturas Reconfiguráveis para Algoritmos de Bioinformática

Jan M. Corrêa (doutorando no ENE)
Alba C. M. A. Melo

Terceiro Seminário Informal(, mas Formal!) do Grupo de Teoria da Computação, 02/12/05

Agenda

- Introdução e Objetivos
- Comparação de Sequências Biológicas
- Arquiteturas Reconfiguráveis
- Arquiteturas Reconfiguráveis para Comparação de Sequências
- Próximos Passos

Introdução

- A comparação de seqüências biológicas é uma das operações mais importantes da biologia computacional.
- Os organismos recém-sequenciados são comparados com organismos já catalogados para determinação das suas funcionalidades e propriedades.
- A cada dia, um grande número de comparações são executadas.

Introdução

- O método exato para comparação de duas sequências biológicas usa programação dinâmica e possui complexidade quadrática de tempo e espaço.
  - Hirschberg reduziu a complexidade de espaço para O(n), ao custo de dobrar o tempo de execução
- Dados o tamanho das sequências comparadas e o número de comparações a serem feitas, o tempo gasto é muito grande.
  - Arquiteturas dedicadas são uma alternativa

Objetivo

- Propor, implementar e avaliar uma arquitetura dedicada em hardware reconfigurável para a comparação de seqüências biológicas usando o método exato de programação dinâmica.

Comparação de Seqüências

- Seqüências de DNA são formada por bases nitrogenadas A, G, C e T.
- A comparação de seqüências é um problema básico da biologia computacional
- Suposição : Se duas seqüências de DNA são similares, possuem funções genéticas similares
- Comparando seqüências pode-se inferir funções e relacionamentos evolutivos

Comparação de Seqüências

- Sendo um alfabeto $\Sigma = \{ A, T, G, C\}$
- Sendo $P \in \Sigma^*$ o padrão da seqüência procurada com $| P | = m$
- Sendo $B \in \Sigma^*$ a seqüência base com $| B | = n$
- *Sendo $k \in R$ o maior erro permitido*
- *Sendo $d : \Sigma^* \times \Sigma^* \rightarrow R$ a função de distância*

Comparação de Seqüências

- O problema do reconhecimento aproximado de seqüências consiste de :
- Dados Σ, T, P, k e $d()$ encontrar todas as posições j em T para as quais existe um i tal que a distância $d(P,Ti..j) <= k$

Comparação de Seqüências

- A função de distância $d(x,y)$ pode ser entendida como o menor número de operações de inserção, remoção e substituição que sejam capazes de transformar a seqüência x na seqüência y.
- Ente problema é semelhante ao do cálculo dos escores do alinhamento

Comparação de Seqüências

- O problema do alinhamento de duas seqüências consiste tentar inferir o grau de relacionamento entre as seqüências através do número mínimo de operações genéticas (inserção, remoção e substituição) necessária para transformar uma seqüência em outra [SET97] .
- Existem alinhamentos de DNA, RNA e proteínas

Alinhamento Global

- Consiste de alinhar duas seqüências por inteiro
- Needleman-Wunsch [NEE70] propuseram um algoritmo baseado em programação dinâmica para o problema do alinhamento global.

Alinhamento Global

- O melhor ou melhores alinhamentos serão aqueles com maior escore
- Durante o processo evolutivo pode ter ocorrido a inserção, remoção ou substituição de bases
- Cada um dos eventos é contabilizado para o cálculo da matriz mxn de programação dinâmica

Alinhamento Global

- Assim existem 4 possibilidades:
  - Inserção de uma base é contabilizada negativamente
  - Remoção de uma base é contabilizada negativamente
  - Substituição de uma base é por outra contabilizada negativamente
  - Coincidência de bases é contabilizada positivamente

Alinhamento Global

- Relação de recorrência para o cálculo dos elementos

$$M(i,j)$$

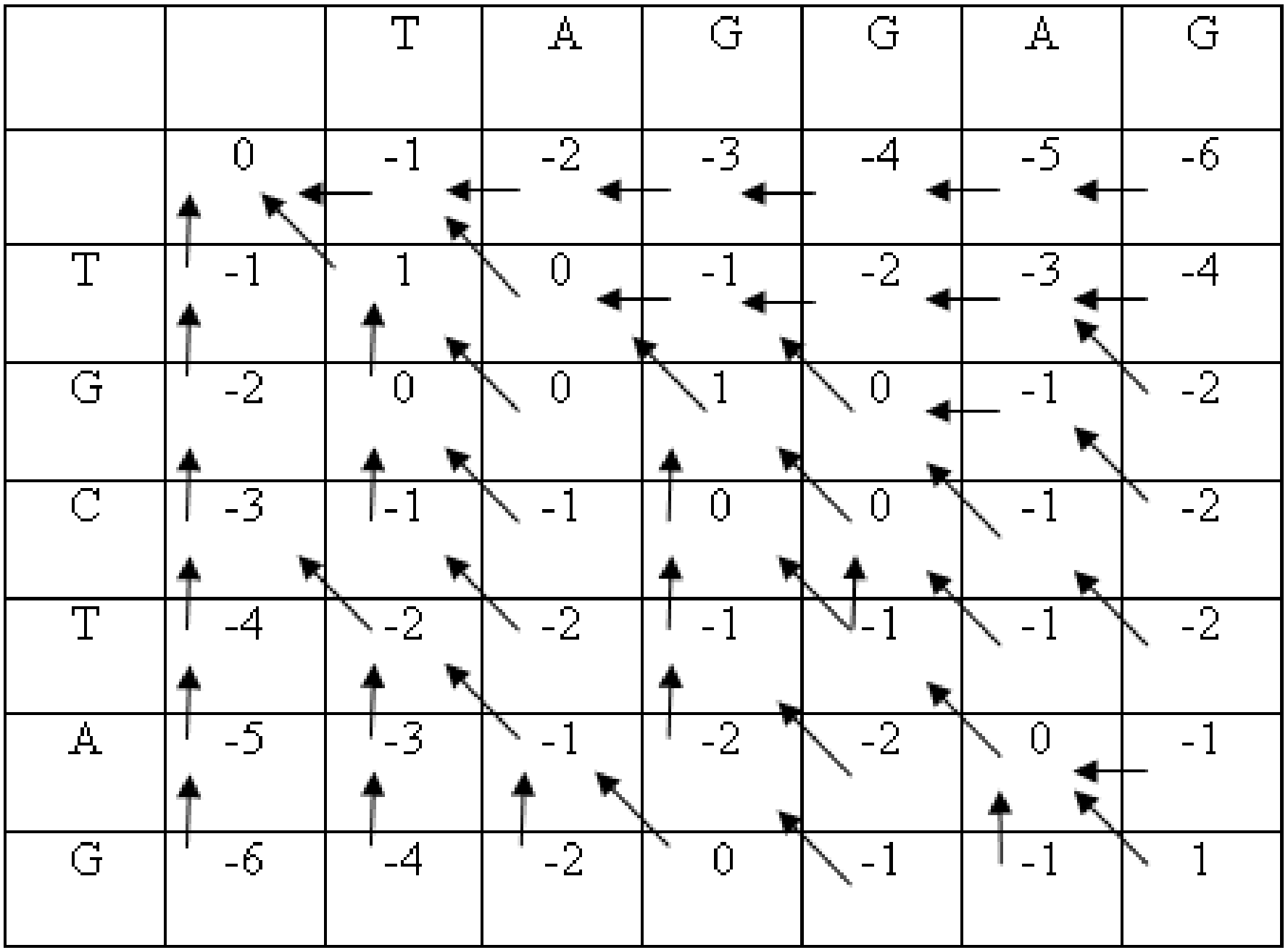
$$\textbf{max} \begin{cases} M(i-1,j) + CI \text{ (custo da inserção)} \\ M(i-1,j-1) + CC \text{ (custo da comparação)} \\ M(i,j-1) + CR \text{ (custo da remoção)} \end{cases}$$

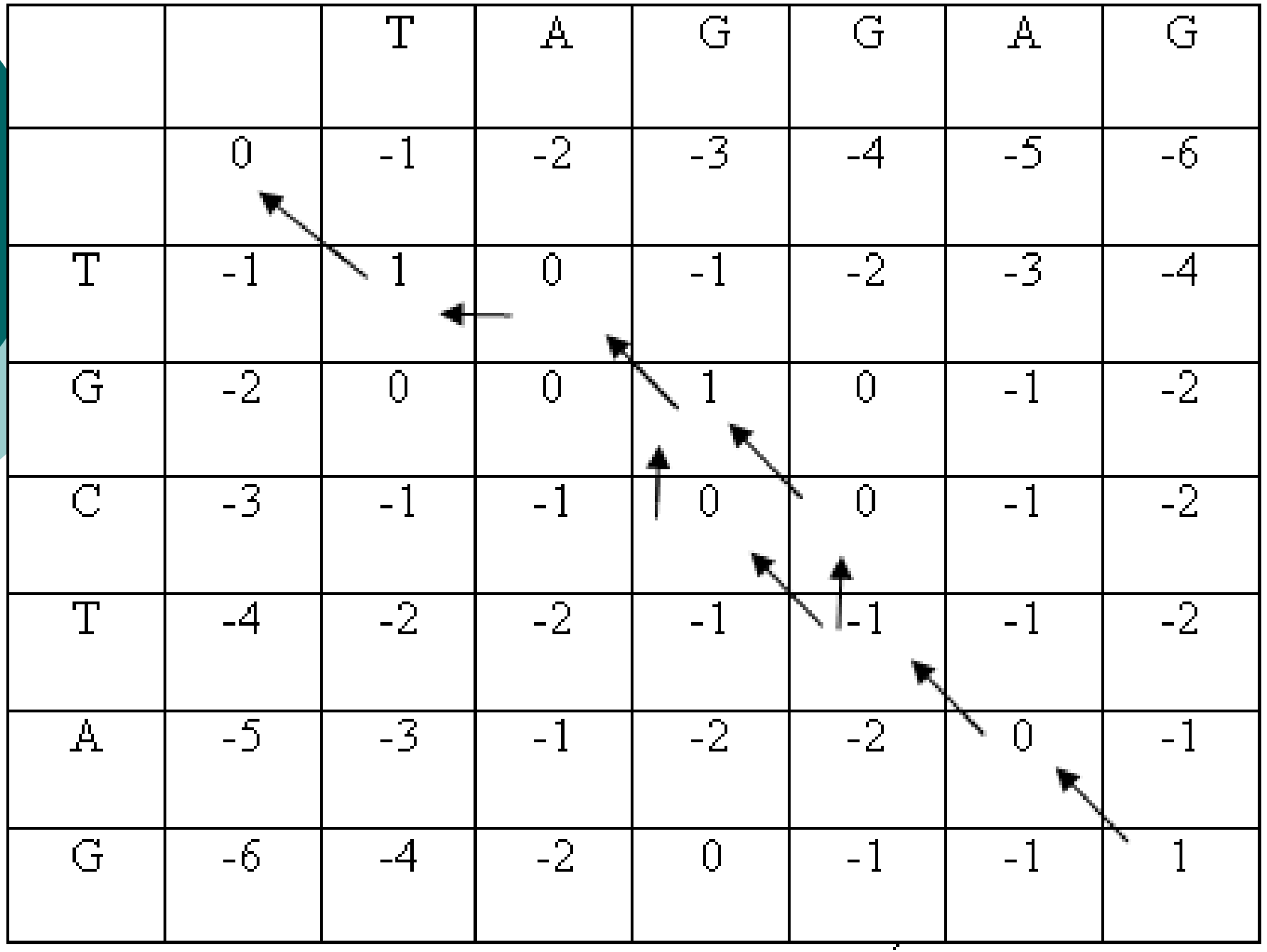
Alinhamento Global

- Assim para o cálculo de cada novo elemento M(i,j) são necessários os valores de M(i,j-1), M(i-1,j) e M(i-1,j-1)
- Após o cálculo da matriz é necessário fazer o caminho reverso de M(m,n) até M(0,0)
- A complexidade do algoritmo no espaço é O(mn) e a complexidade no tempo é também O(mn)

Alinhamento Global

- Exemplo :
  - Seqüências A=TAGGAG e B=TGCTAG
  - A inserção é contabilizada com escore -1
  - A remoção tem escore -1
  - A substituição tem escore -1
  - A coincidência tem escore +1


T A G G A G
0 -1 -2 -3 -4 -5 -6
T -1 1 0 -1 -2 -3 -4
G -2 0 0 1 0 -1 -2
C -3 -1 -1 0 0 -1 -2
T -4 -2 -2 -1 -1 -1 -2
A -5 -3 -1 -2 -2 0 -1
G -6 -4 -2 0 -1 -1 1


T A G G A G
0 -1 -2 -3 -4 -5 -6
T -1 1 0 -1 -2 -3 -4
G -2 0 0 1 0 -1 -2
C -3 -1 -1 0 0 -1 -2
T -4 -2 -2 -1 -1 -1 -2
A -5 -3 -1 -2 -2 0 -1
G -6 -4 -2 0 -1 -1 1

A = T A G – G A G

B = T - G C T A G

1 -1 1 -1 -1 1 1 = 1

(a)

A = T A G G - A G

B = T - G C T A G

1 -1 1 1 -1 1 1 = 1

(b)

Alinhamento Local

- O alinhamento local procura por regiões de alta similaridade dentro das seqüências [SET97]
- Alinhar trechos das seqüências ao invés de alinhar as seqüências inteiras
- interessante do ponto de vista biológico pois mostra as regiões que podem ter se conservado mais ao longo da evolução bem como regiões que podem ter funções semelhantes em genes diferentes

Alinhamento Local

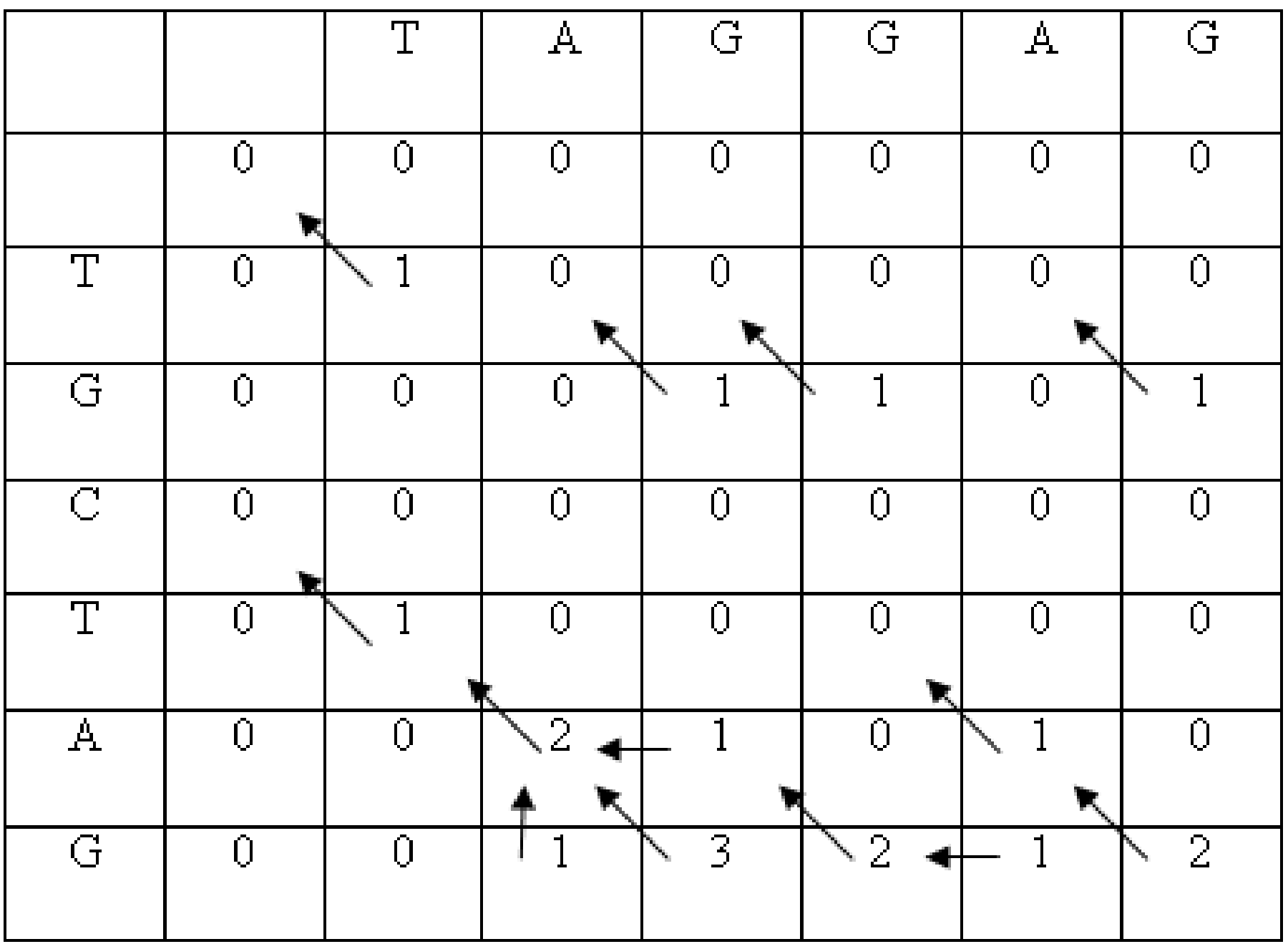
- Um algoritmo de programação dinâmica bastante utilizado é o de Smith-Waterman [SMI81]
- Como o Smith-Waterman tem caráter local, não são permitidos alinhamentos com um número muito grande de inserções, remoções e substituições. Isto é feito atribuindo um valor zero caso o escore se torne negativo
- Complexidades no tempo e espaço de O(mn)

Alinhamento Local

$$M(i,j)\ \textbf{max} \begin{cases} 0 \\ M(i-1,j) + CI \text{ (custo da inserção)} \\ M(i-1,j-1) + CC \text{ (custo da comparação)} \\ M(i,j-1) + CR \text{ (custo da remoção)} \end{cases}$$

Alinhamento Local

- Exemplo :
  - Seqüências A=TAGGAG e B=TGCTAG
  - A inserção é contabilizada com escore -1
  - A remoção tem escore -1
  - A substituição tem escore -1
  - A coincidência tem escore +1


T A G G A G
0 0 0 0 0 0 0
T 0 1 0 0 0 0 0
G 0 0 0 1 1 0 1
C 0 0 0 0 0 0 0
T 0 1 0 0 0 0 0
A 0 0 2 1 0 1 0
G 0 0 1 3 2 1 2

Alinhamento Local

A = **T A G** G A G

B = T G C **T A G**

Arquiteturas Reconfiguráveis

- Dada a relevância e complexidade do problema de comparação de seqüências é interessante aumentar o desempenho do algoritmo através de hardware dedicado
- Arquiteturas dedicadas podem realizar várias operações específicas em paralelo e com alto desempenho

Arquiteturas Reconfiguráveis

- Existe hardware que pode ser programado para funções específicas e depois reprogramado se necessário
- Um tipo de hardware reconfigurável é o FPGA (*Field Programmable Gate Arrays*).

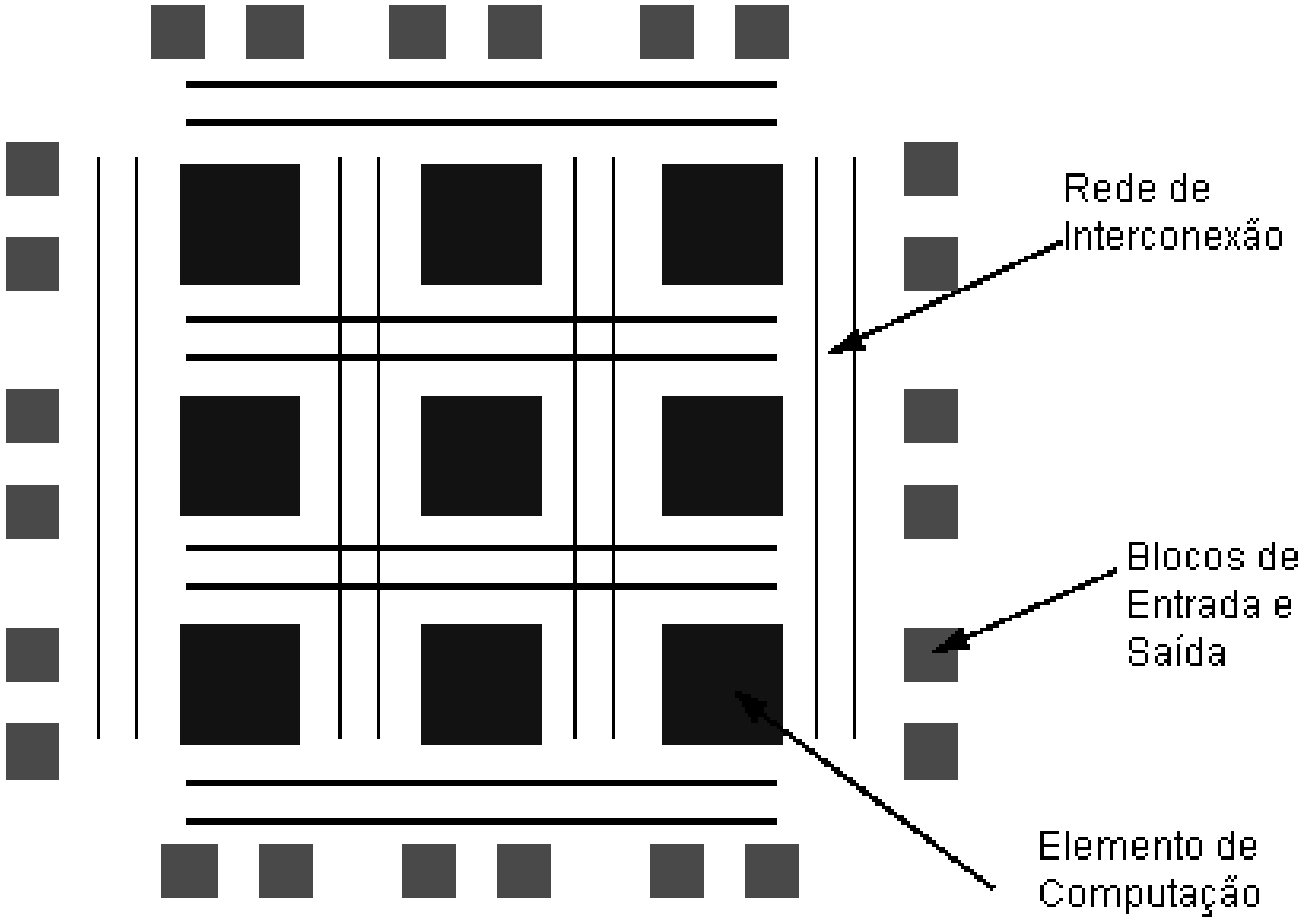
FPGA

- O hardware reconfigurável do FPGA fica em uma placa normalmente colocada no barramento PCI
- O FPGA é controlado por *drivers* específicos
- Normalmente sua capacidade é medida pelo número de transistores ou número de portas lógicas que possui

FPGA

- Em geral possui:
  - Memória RAM
  - Elementos de computação (LUT e registradores)
  - Recursos de roteamento
  - Entrada e Saída (para a placa e RAM interna)

FPGA


Rede de
Interconexão
Blocos de
Entrada e
Saída
Elemento de
Computação

Síntese do FPGA

- A) Especificação do Circuito: o circuito a ser implementado no FPGA é descrito utilizando uma linguagem apropriada como Verilog, VHDL, etc.
- B) Otimização Lógica: Nesta fase as equações lógicas são otimizadas visando uma implementação mais eficiente do circuito

Síntese do FPGA

- C) Mapeamento Tecnológico: Nesta fase é feita a implementação dos circuitos gerados na fase anterior nos elementos de computação presentes no FPGA
- D) Posicionamento: Nesta fase se faz o mapeamento dos elementos de processamento gerados na fase anterior para os blocos físicos onde eles ficarão.

Síntese do FPGA

- E) Roteamento: Esta fase diz respeito à escolha de como os blocos interconexão farão a conexão entre os elementos de processamento mapeados

SystemC

- Linguagem de alto nível para descrição de hardware
- Biblioteca gcc
- Permite modelagem de sistemas complexos
- Maior grau de reutilização do código
- Maior facilidade de detecção de erros
- Síntese não é imediata

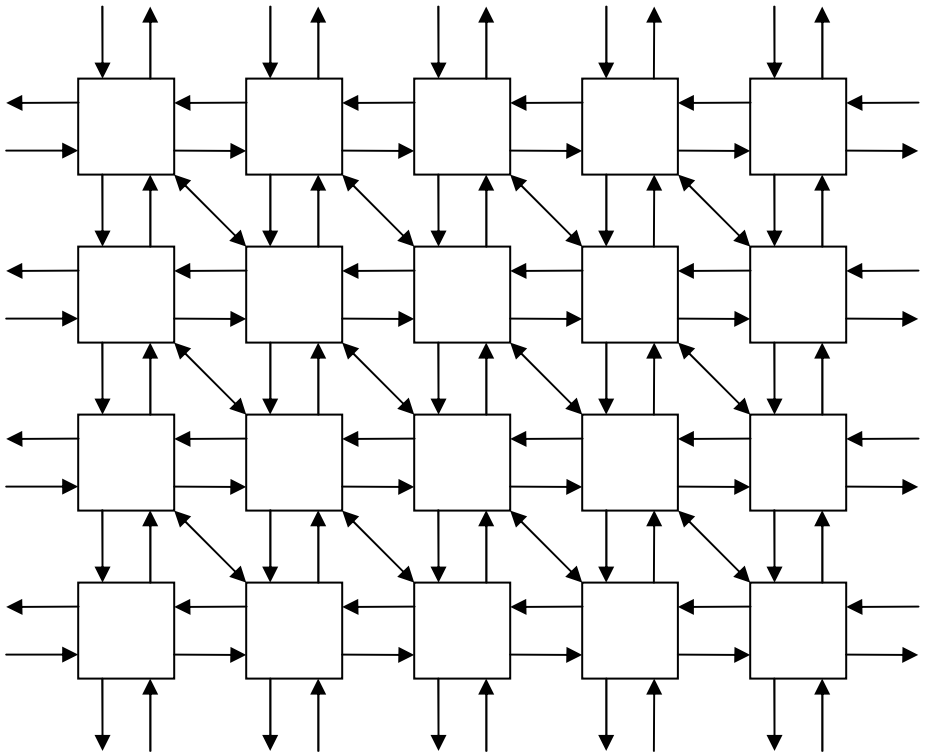
Arquitetura Sistólica

- Permite executar várias operações em paralelo
- Dividir a tarefa entre elementos de processamento simples
- As tarefas devem ser muito parecidas de preferência iguais
- O resultado de um elemento é passado para outro

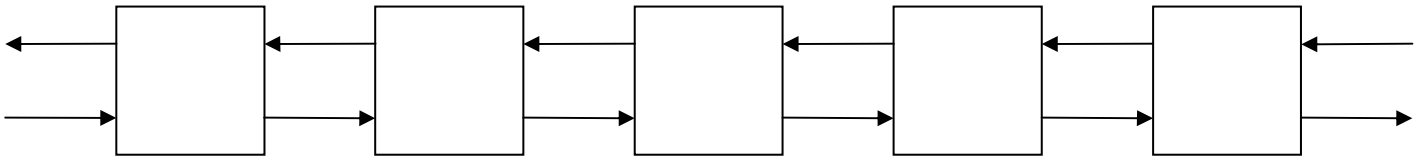
Arquitetura Sistólica

- Preferencialmente deve ser calculado um valor e este deve ser transmitido a cada ciclo de relógio
- Deve haver pouca necessidade de controle dos elementos
- Os elementos não devem trocar muitos dados entre si
- Os elementos devem fazer pouca entrada e saída

Matriz Sistólica

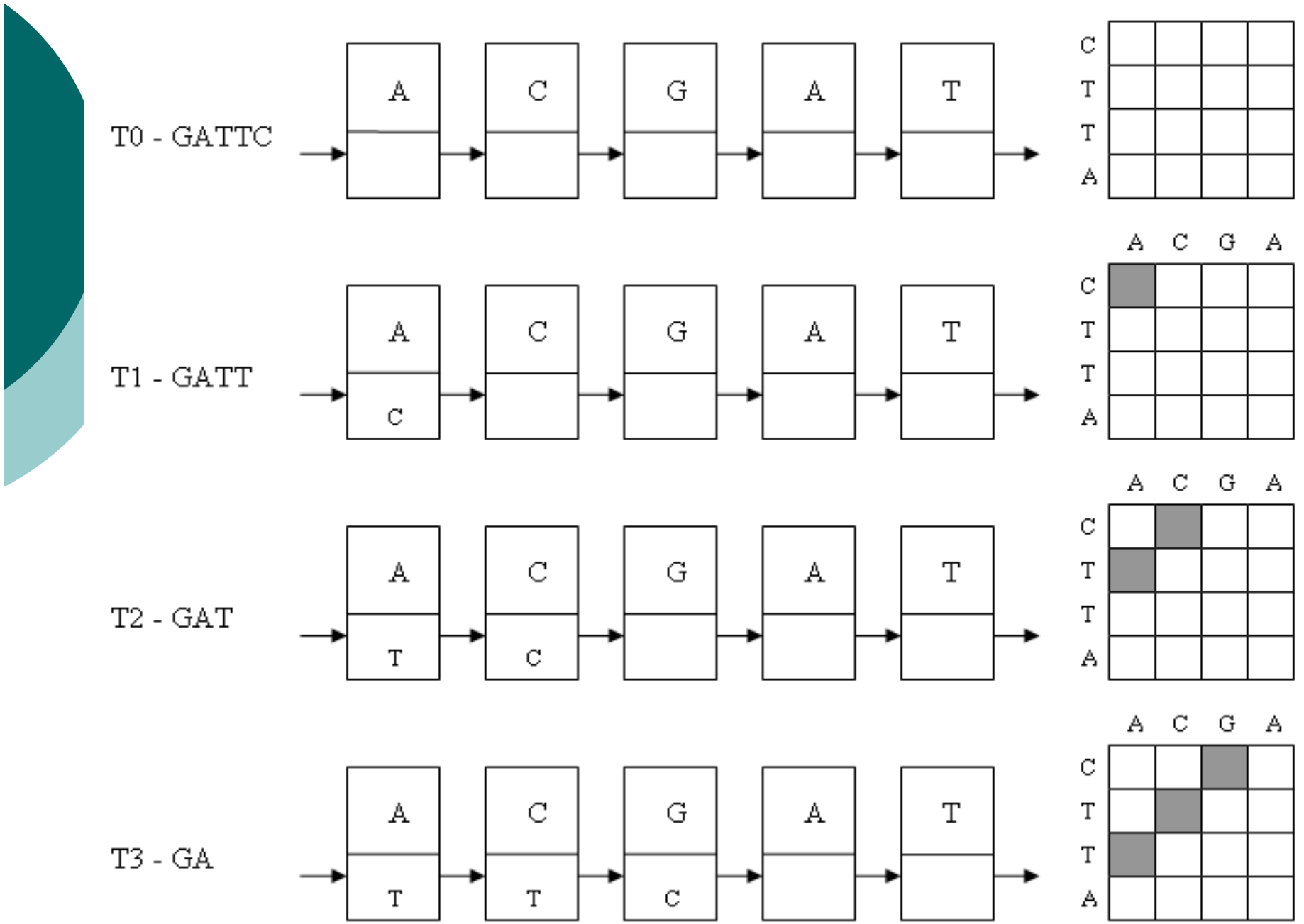


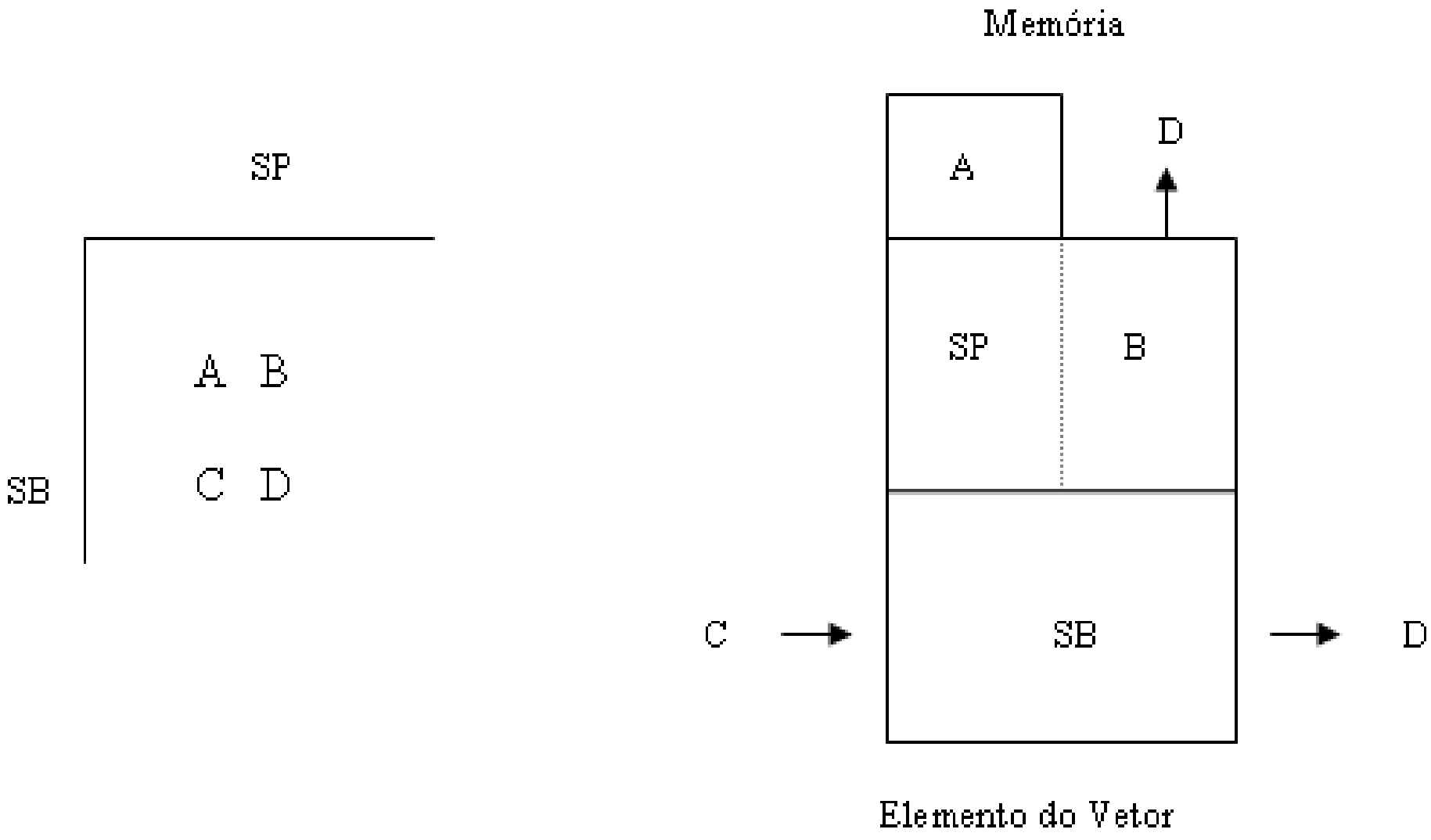
Vetor Sistólico

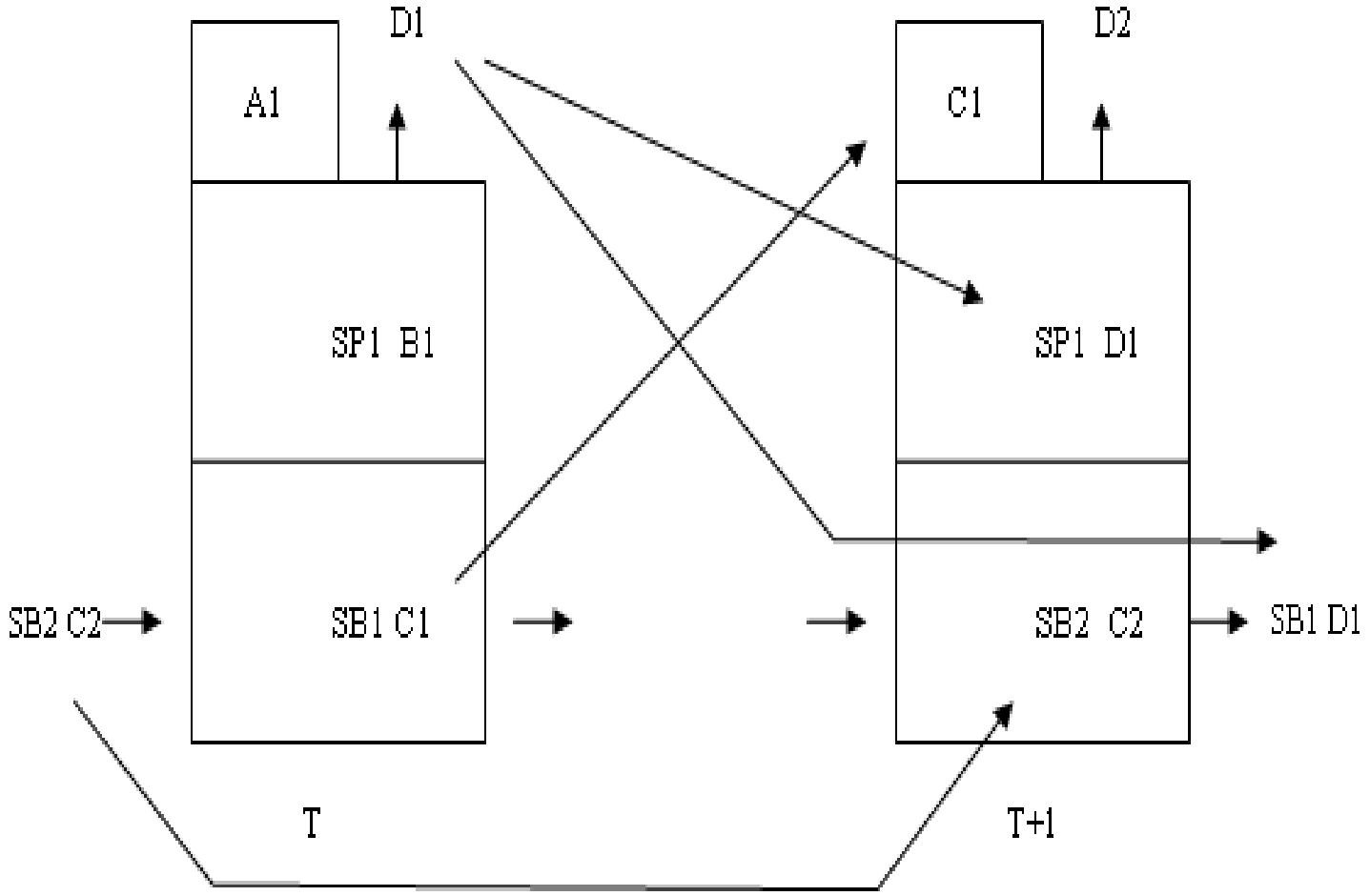


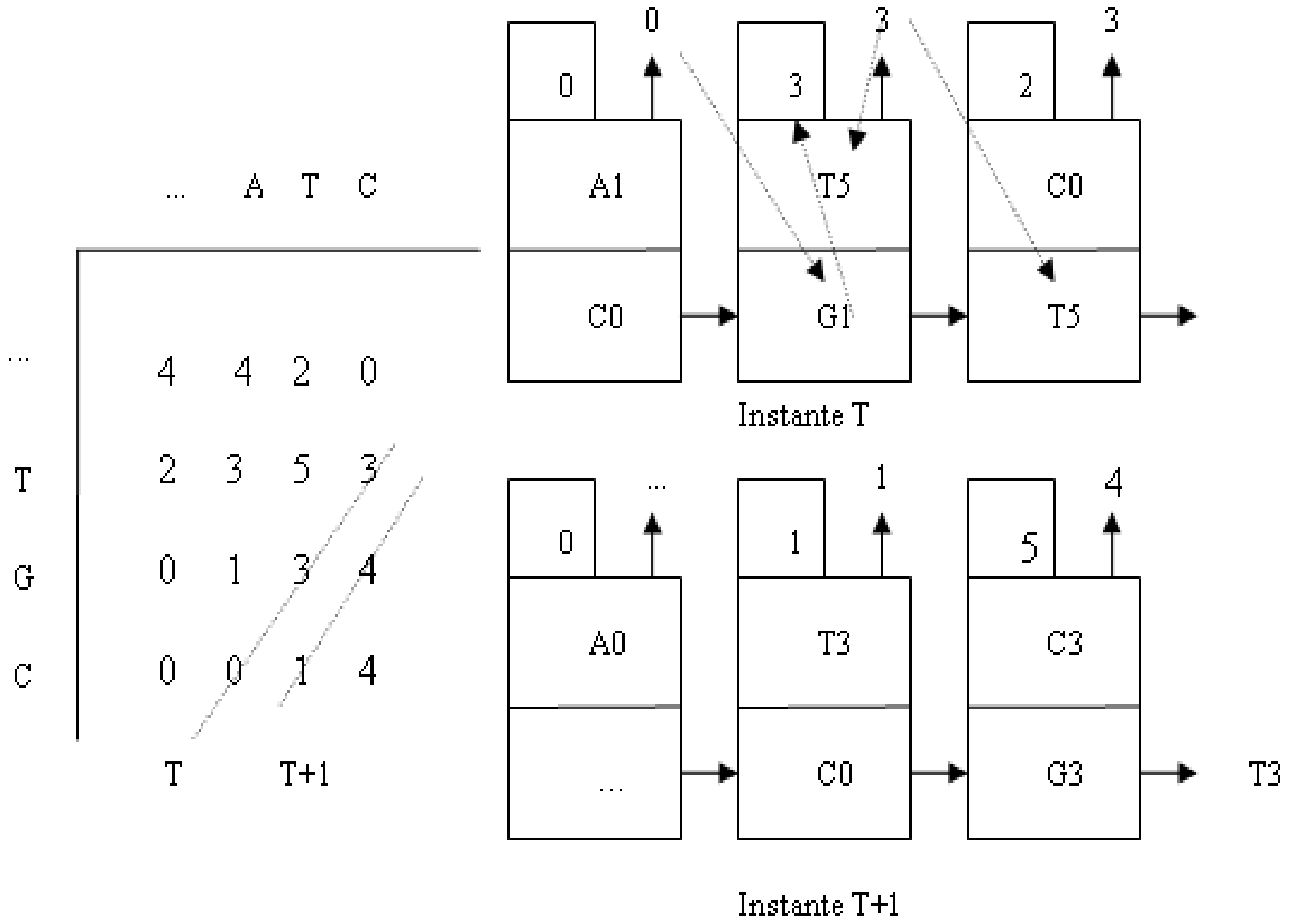
Arquitetura para Comparação de Seqüências

- Mapeamento do Smith-Waterman para um vetor sistólico
- Uma seqüência é colocada no vetor
- A outra flui pelo vetor
- O cálculo da matriz de similaridade é feita através das anti-diagonais


T0 - GATTC
A
C
G
A
T
C
T
T
A
T1 - GATT
A
C
G
A
T
C
A C G A
C
T
T
A
T2 - GAT
A
C
G
A
T
T
C
A C G A
C
T
T
A
T3 - GA
A
C
G
A
T
T
T
C
A C G A
C
T
T
A


SP
A B
SB
C D
Memória
A
D
SP
B
C
SB
D
Elemento do Vetor


A1
D1
C1
D2
SP1 B1
SP1 D1
SB2 C2
SB1 C1
SB2 C2
SB1 D1
T
T+1


... A T C
... 4 4 2 0
T 2 3 5 3
G 0 1 3 4
C 0 0 1 4
T T+1
0
3
3
0
3
2
A1
T5
C0
C0
G1
T5
Instante T
...
1
4
0
1
5
A0
T3
C3
...
C0
G3
T3
Instante T+1

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- Foram analisadas várias implementações de arquiteturas reconfiguráveis (FPGA) para processamento de algoritmos de bioinformática
- Aqui serão mostradas implementações para comparação de seqüências

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- FPGA não possui muita memória interna
- Em alguns casos foram implementadas políticas de particionamento
- Métrica muito utilizada MCUPS (million cell updates per second)
- Cada atualização é o cálculo de uma posição da matriz, pode ser calculado multiplicando-se o número de elementos de processamento (PE) pela freqüência de relógio (clock) na qual elas operam

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- Esta medida não leva em consideração os tempos necessários para fazer as transferências da RAM para os elementos de processamento
- Comparação difícil pois vários modelos de FPGA, capacidades e arquiteturas testadas.

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- Muitos implementaram quebras de seqüência de base
- Importante para aplicações reais onde as seqüências são grandes
- A maioria das implementações do Smith-Waterman calculou apenas as matrizes com os escores e não realizaram a etapa de encontrar os melhores alinhamentos.

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- Em [LAV98] houve a implementação de algoritmo de Smith-Waterman usando uma placa SAMBA (Systolic Accelerator for Molecular Biological Applications)
- Em uma comparação entre o SAMBA e uma DEC Alpha 150 MHz para procurar em um banco de dados de 21210389 aminoácidos e seqüência procurada de 3000, conseguiu um tempo de 3:20 minutos e a DEC Alpha de 280:00 sendo portanto 83 vezes mais rápida.

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- [WES03] se apresenta uma implementação do algoritmo Smith-Waterman. Utilizou um FPGA FPX ligado ao barramento ATPI/IDE para acessar o disco diretamento
- Procurando um padrão de 38 bytes. Em comparação a um Pentium III 933 MHz teve um Speed-UP de 50.1 em tempo e de 123.8 em CUPs

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- Em [HOA92] se implementou o algoritmo de Needleman-Wunsch em uma placa Xilinx XC3090
- Conseguiu speedup de 100 (tempo) em comparação a um Cray-2 para fazer 10000 alinhamentos em seqüências de tamanho 100.

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- [PUT03], a implementação do Smith-Waterman usando um FPGA Virtex II6000 conseguiu um speedup de 500 sobre uma placa Timelogic em CUPs
- Usando reconfiguração dinâmica conseguiu 1,3 trilhões de CUPs

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- [YAM02] implementou o Smith-Waterman em um FPGA Xilinx XCV2000E
- Para alinhar uma seqüência de 2048 com um banco de dados de 64 MB demorou 34 segundos o que é quase 330 vezes mais rápido do que um PentiumIII de 1GHz

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- A etapa de cálculo dos escores é muito mais demorada que a de encontrar os alinhamentos.
- Assim não compensaria o esforço da sua implementação em FPGA já que o computador hospedeiro poderia executá-la em pouco tempo
- 1024 elementos e banco de dados de 4096 elementos demorou 0.007 segundos enquanto o Pentium III fez em 0.35 segundos

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- Em [GRA01] foi mostrada a implementação do Smith-Waterman em uma placa Kestrel
- Para procurar 512 bytes em 10 MB teve um speedup de 20 vezes em relação a uma DEC Alpha 433 MHz

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- Em [TIM04] houve a implementação do Smith-Waterman em Verilog com FGPA Virtex II XC2V6000
- Particionamento colocando até 4 bases da seqüência procurada por elemento
- Para comparações de 1512 bases com um Pentium 4 1.6 GHz o speedup chegou a 170 em CUPs

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- A reconfiguração dinâmica como em [GUC02] feita com um Virtex XC2V6000 permite colocar os elementos das seqüências bem como os escores diretamente em hardware aumentando a velocidade
- Demanda o tempo da reconfiguração
- Conseguiu um speedup de 65 contra uma placa Splash-II em CUPs

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- Para melhorar o desempenho deve-se diminuir a necessidade de memória e da entrada e saída
- Uma saída e diminuir a quantidade de memória utilizada como em [CAR03] onde houve a redução de até 80% da matriz

Artigo	FPGA	Sequencias	Quebra de Sequencia	GCUPS	"Speed-Up"	Tipo	Recon. Din.	Alinhamento
[LAV98]	SAMBA	3K x 2.1M	Sim	Nao Disp.	83	DEC Alpha 150 MHz	Não	Não
[GUC02]	Virtex XC2V6000	Não Disp.	Sim	3,220	65	Splash-2	Sim	Não
[WOE02]	XC2V1000	ND x MBs .	Sim	389	5000	Pentium III 1.4 GHz	Sim	Não
[MAR03]	Xilinx XV1000	24x 2M	Não	Não Disp.	5.6	Pentium III 1 GHz	Não	Não
[LAV96]	Splash-2	Não Disp.	Não	5	1	VLSI prog.	Não	Não
[WES03]	FPX	38 x MBs	Não	3.8	50.1	Pentium-III 933 MHz	Não	Não
[HOA92]	Splash Xilinx XC3090	100 x MBs	Não	Não Disp.	100	Cray-2	Não	Sim
[MOS98]	Xilinx XC4000	ND x MBs	Sim	0.06	100	Sun Spark	Não	Não
[Yu03]	Xilinx XCV1000	Não Disp.	Não	0.8	?	Não Conclusivo	Não	Não
[PUT03]	OSIRIS	ND x 48MB	Sim	1260	500	Timelogic	Sim	Não
[YAM02]	Xilinx XCV2000E	2K x 64MB	Sim	Não Disp.	330	PentiumIII de 1GHz	Não	Sim
[GRA01]	Kestrel	512x10M	Sim	Não Disp.	20	DEC Alpha 433 MHz.	Não	Não
[TIM04]	Virtex II XC2V6000	1.5k x ND	Sim	1,4	170	Pentium 4 1.6 GHz	Não	Não

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- Em aplicações reais as seqüências tem vários KBs
- Se duas seqüências de 20 KB fossem comparadas geraria uma matriz de 400 MB!
- Isto é maior que a capacidade da maioria dos FPGAs

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- Necessidade de transmitir a matriz para a memória principal
- Os elementos da matriz precisam ser enviados por um barramento.
- Em alguns casos o barramento é o PCI que é muito mais lento que o barramento da memória principal

Arquiteturas Reconfiguráveis para Comparação de Seqüências

- A quebra da seqüência procurada também é importante para aplicações reais onde elas são grandes e não cabem no FPGA
- Fazer isto de forma eficiente é difícil
- Isto não foi explorado nas aplicações

Próximos Passos

- Criar uma arquitetura sistólica eficiente para aplicações reais que utilize o paralelismo da arquitetura de forma eficiente utilizando pouca memória e realizando poucas operações de entrada e saída
- Implementar esta arquitetura em FPGA e comparar o resultado com outras arquiteturas

Referências

- [CAR03] Carvalho, Luís Gustavo de Aquino, Uma abordagem em hardware para algoritmos de comparação de seqüências baseados em programação dinâmica, Universidade de Brasília, 2003
- [GUC02] Steven A. Guccione, Eric Keller, Gene Matching Using JBits, Field-Programmable Logic and Applications, Reconfigurable Computing Is Going Mainstream, 12th International Conference, FPL 2002
- [HOA92], Dzung T. Hoang, FPGA Implementation of Systolic Sequence Alignment, Field-Programmable Gate Arrays: Architectures and Tools for Rapid Prototyping, H. Grunbacher and R. W. Hartenstein, eds., Berlin: Springer-Verlag, 1992, pp. 183-191
- [KNO04] Knowles, Greg, Gardner-Stephen, Paul, A New Hardware Architecture for Genomic Sequence Alignment, 3rd International IEEE Computer Society Computational Systems Bioinformatics Conference (CSB 2004), 16-19 August 2004, Stanford, CA, USA. IEEE Computer Society 2004
- [LAV96] D. Lavenier, Dedicated Hardware for Biological Sequence Comparison, Journal of Universal Computer Science, 2 (2) 1996.

Referências

- [LAV98] D. Lavenier, Speeding up genome computations with a systolic accelerator, *SIAM news,* 31 (8) 1998.
- [MAR03] A.Marongiu, P.Palazzari, V.Rosato, PROSIDIS: a Special Purpose Processor for PROtein SImilarity DIScovery,Second IEEE International Workshop on High Performance Computational Biology 2003 Nice, França
- [MOS98] Mosanya, Emeka, "A Reconfigurable Processor for Biomolecular Sequence Processing", tese de doutorado, Ecole Polytechnique Fédérale de Lausanne, 1998.
- [NEE70] Needleman S. B. e Wunsch C. D., A General Method Applicable to the Search for Similarities in the Amino-Acid Sequence of Two Proteins, Journal of Molecular Biology, 48, pp. 443–453, 1970
- [PUT03] Puttegowda, K.; Worek, W.; Pappas, N.; Dandapani, A.; Athanas, P.; Dickerman, A., A run-time reconfigurable system for gene-sequence searching, Proceedings. 16th International Conference on VLSI Design, 2003
- [SET97] Setubal, J. e Meidanis, J., Introduction to Computational Molecular Biology. PWS Publishing Company, Boston. 1997
- [Smi81] Smith T.F. and Waterman M.S. (1981) Identification of common molecular sub-sequences. Journal of Molecular Biology, 147 (1) 195-197. PubMed.

Referências

- [TIM04] Timothy, Oliver, Bertil Schmidt, High Performance Biosequence Database Scanning on Reconfigurable Platforms, Third IEEE International Workshop on High Performance Computational Biology (HiCOMB) Santa Fé, Estados Unidos
- [WOE02] William J. Worek dissertação de mestrado, Virginia Polytechnic Institute and State University, 2002
- [YAM02] Yamaguchi, Y., Maruyama, T., Konagaya, A., High Speed Homology Search with FPGAs,. Pacific Symposium on Biocomputing (PSB 2002) (Kaua'I USA) 2002
- [Yu03] Yu C.W., Kwong, K.H., Lee, K.H., Leong, P. H. W., A Smith-WatermanSystolic Cell, 13 th Int. Conference on Field-Programmable Logic and Applications, Springer-Verlag, Lisboa, Portugal 2003