Processamento Paralelo Aplicado a Métodos Filogenéticos Comparativos

Elias Batista Ferreira
e-mail:eliasbf@gmail.com

Orientador: Wellington Santos Martins
Instituto de Informática

04 de outubro de 2012

Sumário

Sumário

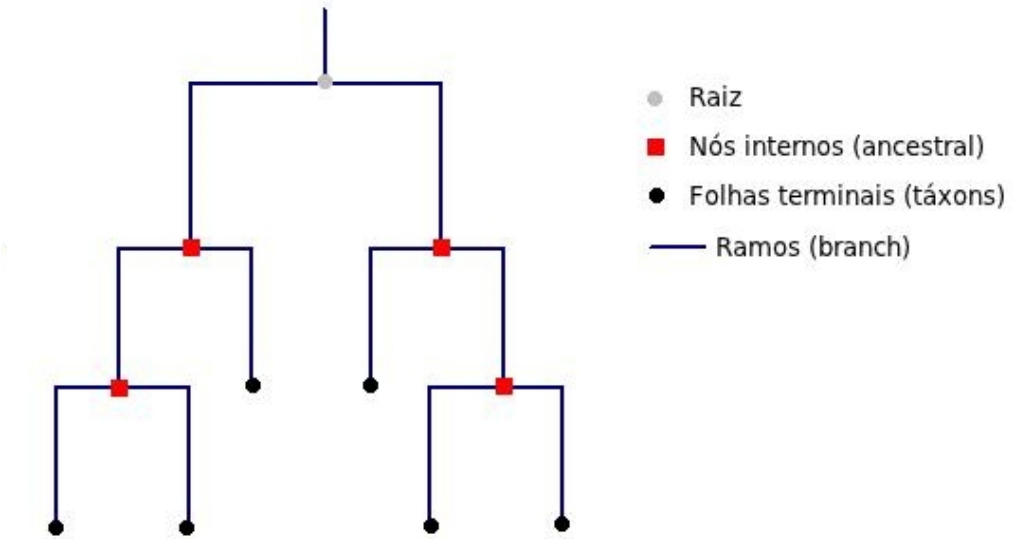
Conceitos

- **Filogenia**: é o termo utilizado para definir hipóteses de relações evolutivas, ou seja, relações filogênicas, de um grupo de organismos.
- **Métodos Filogenéticos Comparativos**: utiliza as relações filogenéticas como base para analisar estatisticamente os padrões de variação e/ou covariação de caracteres morfológicos, ecológicos, comportamentais, etc.

Algumas analises filogenéticas comparativas dependem de procedimentos de simulação, que utilizam grande numero de árvores filogenéticas para estimar as correlações evolucionarias.

Árvore filogenética

As relações filogenéticas são representadas graficamente como árvores filogenéticas, também designada por Árvore da Vida



Motivação

- Devido ao fardo computacional de processamento de centenas de milhares de árvores, a menos que este procedimento seja eficazmente aplicado, as análises são de aplicabilidade limitada.
- Outros trabalhos sobre análises filogenéticas computacionais têm se concentrado em inferir árvores filogenéticas (Máxima Parcimônia, Máxima Verossimilhança e análise Bayesiana).
- Aqui fazemos uso de árvores filogenéticas disponíveis para realizar estudos comparativos

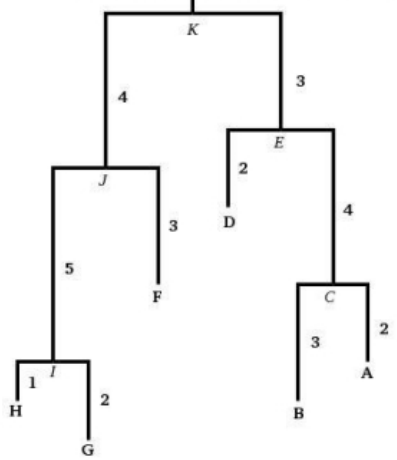
Computação paralela

- Consiste na utilização de múltiplos núcleos de processamento para executar partes diferentes de um mesmo programa.
- Principal objetivo é a busca por melhor desempenho.
- Algumas soluções possuem alto custo; quando utilizados supercomputadores tradicionais.
- Soluções com GP-GPU tornam o paralelismo facilmente acessível e com grande poder de processamento.

Sumário

Passos

1 Árvore consenso



Espécies faltantes

espécie	mdcc
X	K
Y	J

2 Matriz Patrística

	0	1	2	3	4	5
0	0	x1	x2	x3	x4	x5
1	x1	0	x6	x7	x8	x9
2	x2	x6	0	x10	x11	x12
3	x3	x7	x10	0	x13	x14
4	x4	x8	x11	x13	0	x15
5	x5	x9	x12	x14	x15	0

3 I de Moran por classe

Classe	I de Moran
0,00 – 1,35	0,75
1,36 – 2,70	-1,00
2,71 – 3,05	0,10
3,06 – 4,40	1,00

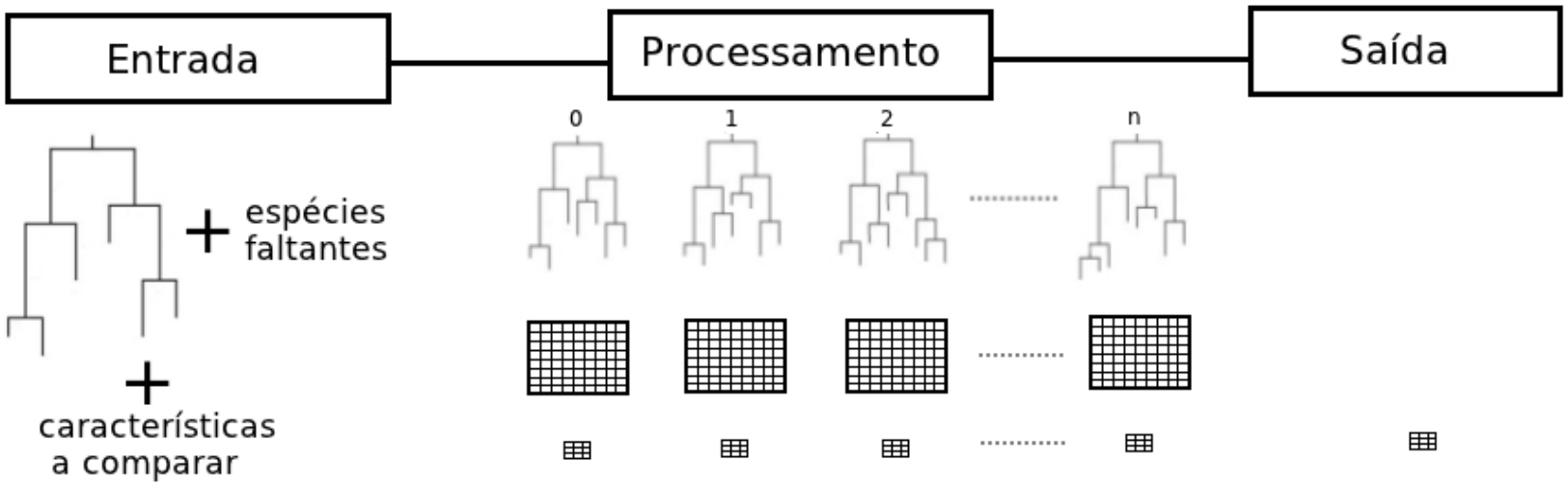
Passos

- **Árvore consenso**: árvore filogenética que os biólogos acreditam estar correta. No entanto, em sua maioria são incompletas.
- **Espécies faltantes/perdidas**: espécies sobre as quais não se conhece o ponto exato para inserção das mesmas.
- **Matriz de distâncias patrística**: distância filogenética entre os pares de espécies. Uma distância patrística é a soma de todos os comprimentos de ramos entre duas espécies pertencentes a árvore.
- **I de Moran**: permite verificar a correlação evolutiva de uma determinada característica da espécie.

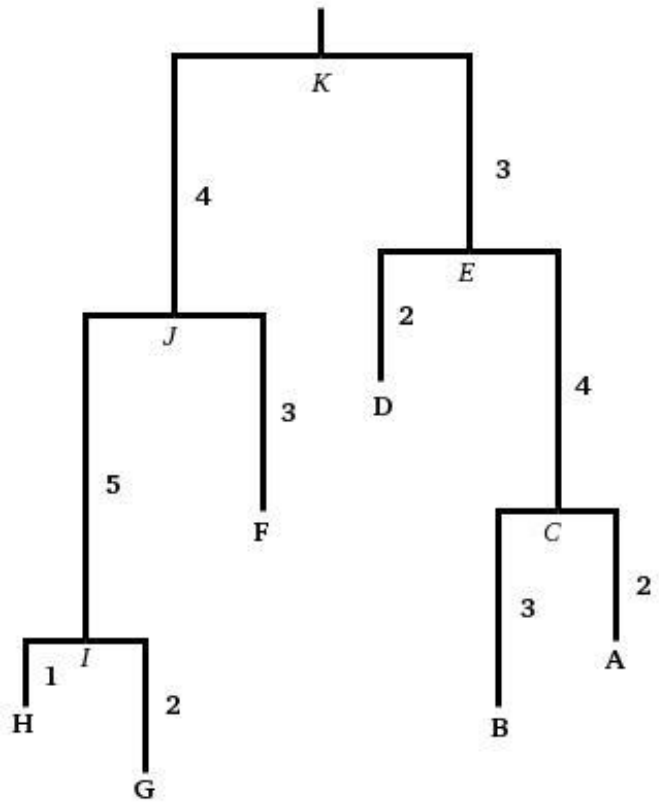
Desafios

- Devemos gerar milhares de árvores com desempenho superior aos programas atuais.
- **Estratégia:** utilizar paralelismo para: adicionar espécies, calcular as matrizes de distância patrísticas e calcular o *Índice de Moran*.
- Além disso, queremos explorar paralelismo ao calcular uma única matriz; e, também, ao calcular o *I de Moran* para cada matriz.
- Utilização de arquiteturas de computador paralelas contendo dezenas (ou centenas) de núcleos, por exemplo, GPU (Unidade de Processamento Gráfico)
- Isso geralmente requer novos algoritmos e uso de uma linguagem de programação especial, *e.g.*, CUDA (Compute Unified Device Architecture) da NVIDIA.

Visão geral


Entrada
Processamento
Saída
+ espécies faltantes
+
características a comparar
0
1
2
n

Dados de entrada: árvore filogenética no formato Newick




```
(((A:2,B:3)C:4,D:2)E:3,(F:3,(G:2,H:1)I:5)J:4)K;
```

Dados de entrada: espécies faltantes e características utilizadas para comparação

espécie	mdcc
X	K
Y	J

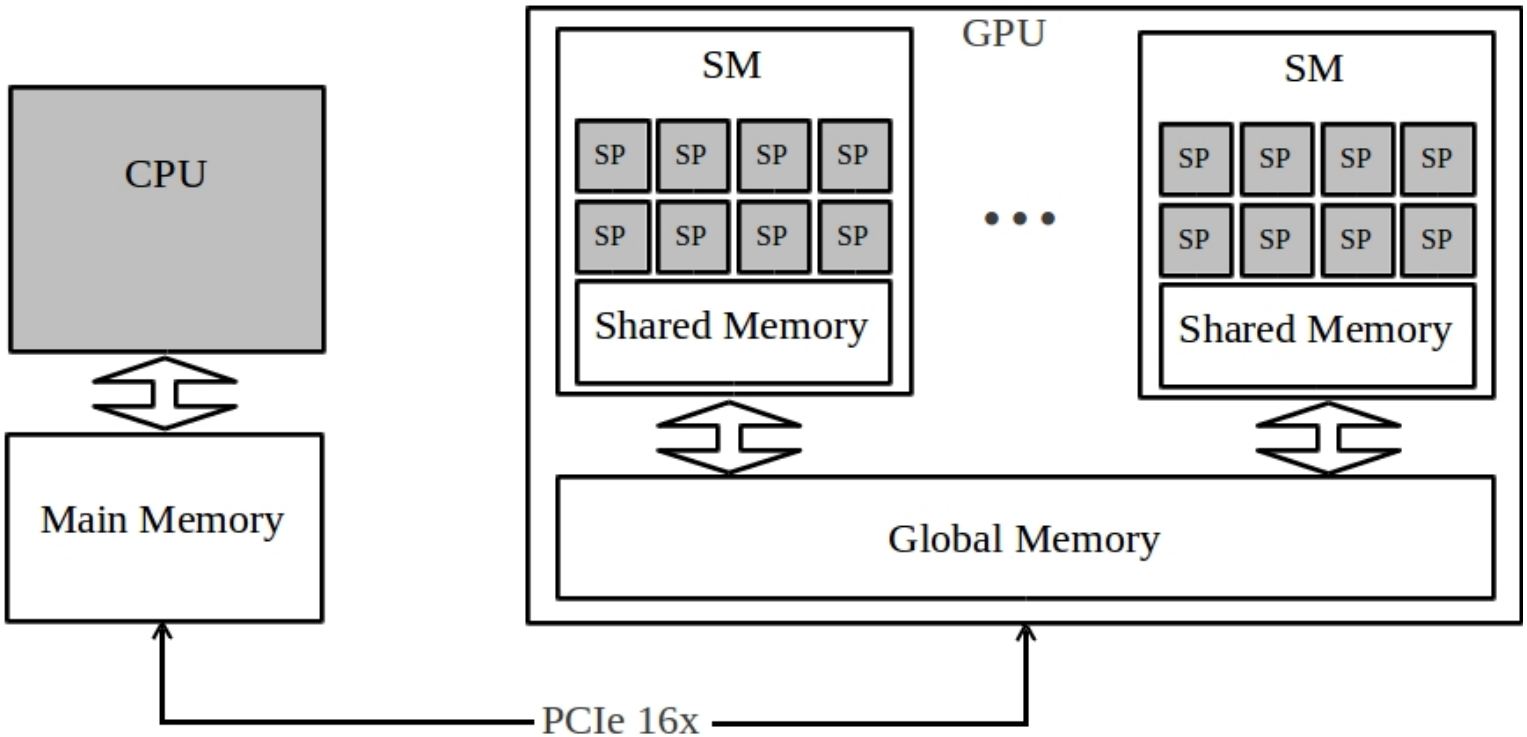
Espécies faltantes/perdidas

name	body
A	3.3
B	2.4
D	6.0
F	2.1
G	3.4
H	5.6
X	4.3
Y	5.5

Relação de características das espécies

Sumário

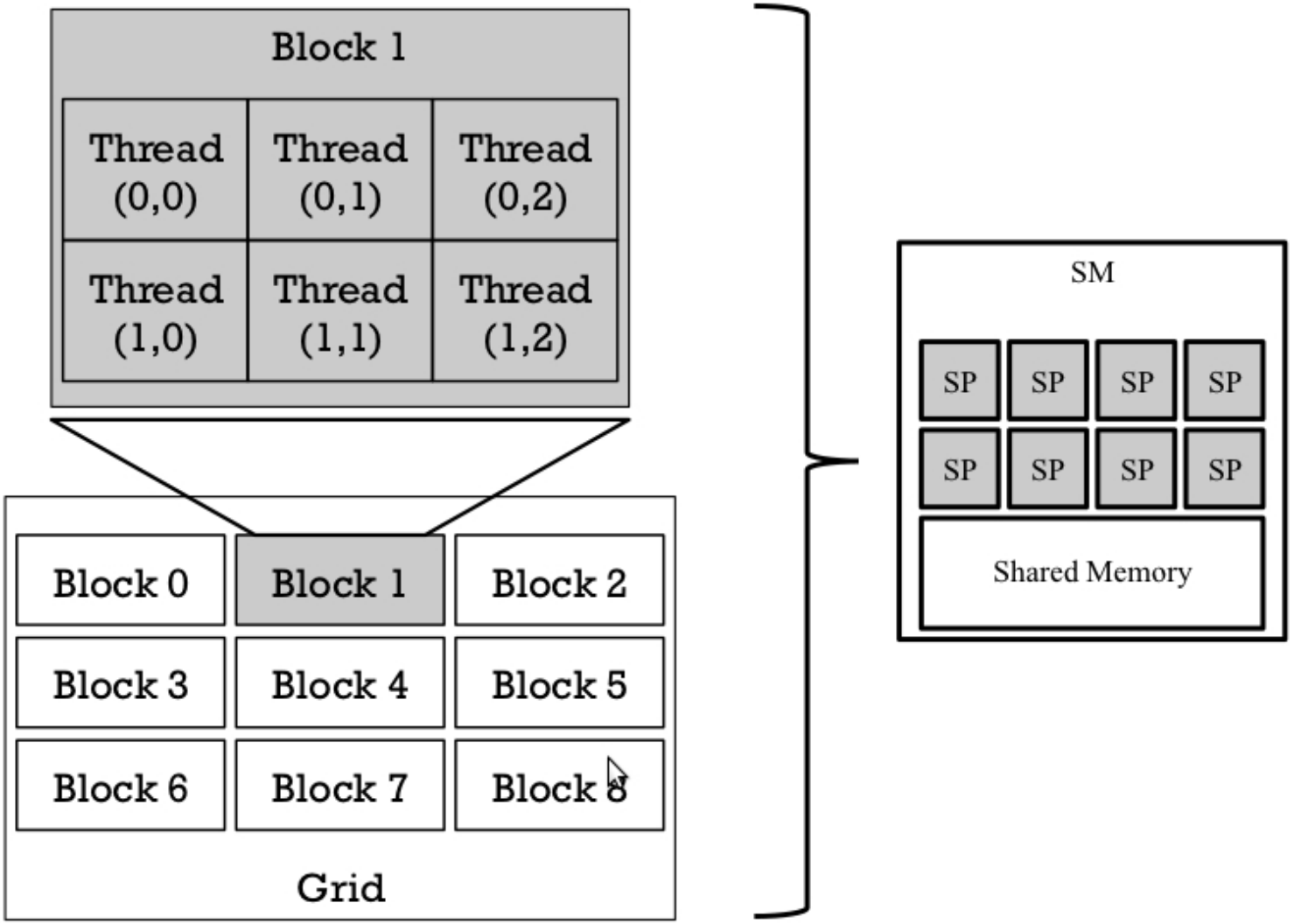
Arquitetura GPU NVIDIA



Linguagem CUDA

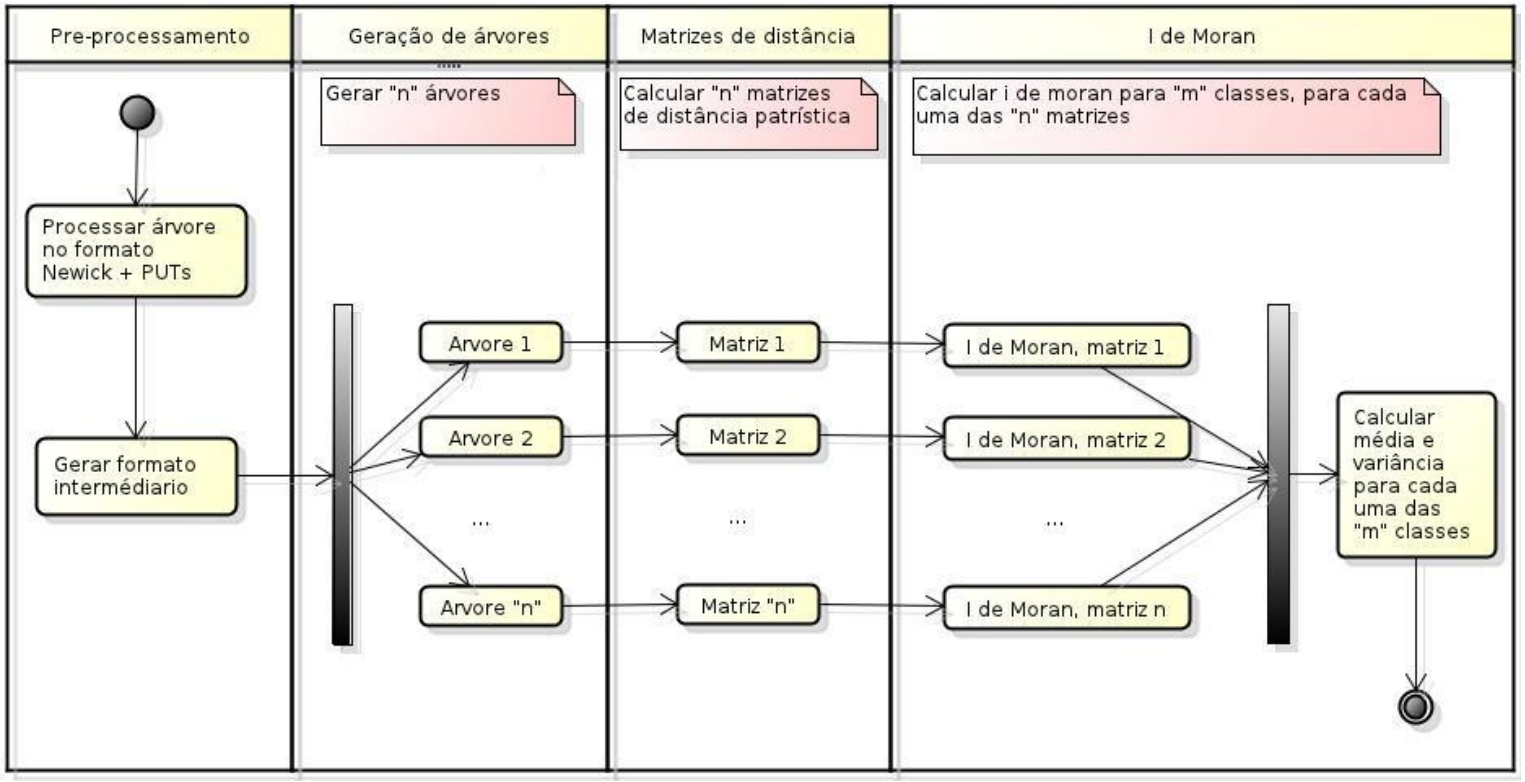
- CUDA, sigla para Compute Unified Device Architecture. É uma extensão para a linguagem de programação C, a qual possibilita o uso de computação paralela sobre GPU.
- Permite que programadores utilizem todo o poder de processamento de suas placas de vídeo (GPUs) em algoritmos otimizados ao uso em paralelismo
- **Kernel:** função geralmente escrita em C para CUDA. Seu código é executado por cada *thread*.
- **Threads e Blocos**: as threads são organizadas em bloco. Um bloco é um arranjo de *threads*.
- **Grid**: blocos são organizados em grids.

Mapeando *threads* para GPU



Sumário

Etapas da solução



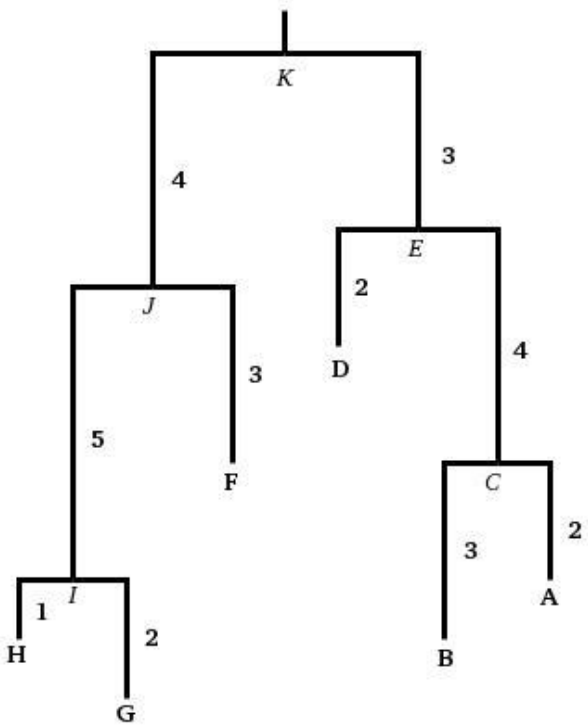
Visão geral

Objetivo

Preparação dos dados de entrada em uma estrutura adequada para manipulação pelo sistema computacional em GP-GPU. Os dados de entrada são:

- **Árvore filogenética**, parcialmente conhecida e representada no formato Newick. Normalmente construída à partir de dados moleculares.
- **PUT** (phylogenetically uncertain taxa), que são conjunto de espécies faltantes na árvore de entrada.
- **Traits**: são características das espécies que se deseja comparar. São utilizados nos cálculos de correlação filogenética.

Mapeamento das entradas para uma estrutura de dados do tipo vetor



	espécies iniciais						novas espécies			novos ancestrais		ancestrais iniciais				
indice	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
espécie	A	B	D	G	H	F	*X*	*Y*	-	*?*	*?*	I	J	C	E	K
pai	13	13	14	11	11	12	*15*	*12*	-	*?*	*?*	12	15	14	15	-1
filho esquerda	-2	-2	-2	-2	-2	-2	*-2*	*-2*	-	*?*	*?*	3	11	0	13	14
filho direita	-2	-2	-2	-2	-2	-2	*-2*	*-2*	-	*?*	*?*	4	5	1	2	12
comprimento do ramo	2	3	2	2	1	3	*?*	*?*	-	*?*	*?*	5	4	4	3	0
característica (traits)	3,3	2,4	6,0	3,4	5,6	2,1	*4,3*	*5,5*	-	*0,0*	*0,0*	0,0	0,0	0,0	0,0	0,0

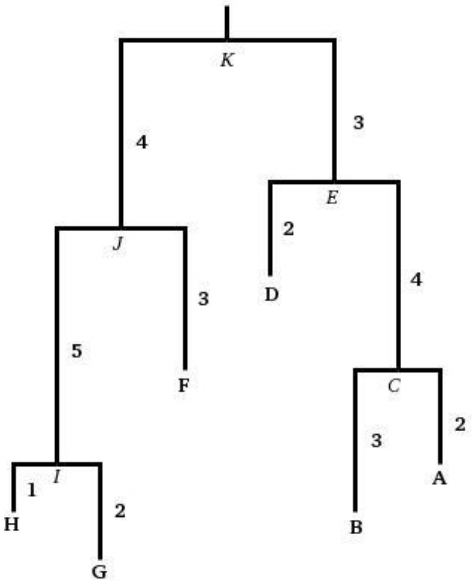
Visão geral

Processo de adição de espécies

- Receber as estruturas de dados com as árvores
- Lançar "*n*" *threads* para computar as "*n*" árvores
- Cada *thread* deve identificar a faixa de dados correspondente a árvore que irá manipular
- Inserir as espécies aleatoriamente à partir do MDCC (most derived consensus clade)
- Manter a árvore binária (dicotômica)
- Atualizar as estruturas de dados (vetores) para refletir as relações a cada espécie inserida

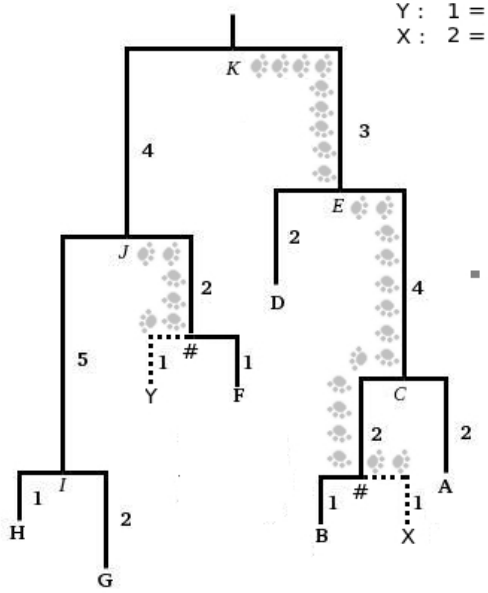
Simulação para adição dos PUTs

árvore original com 6 espécies



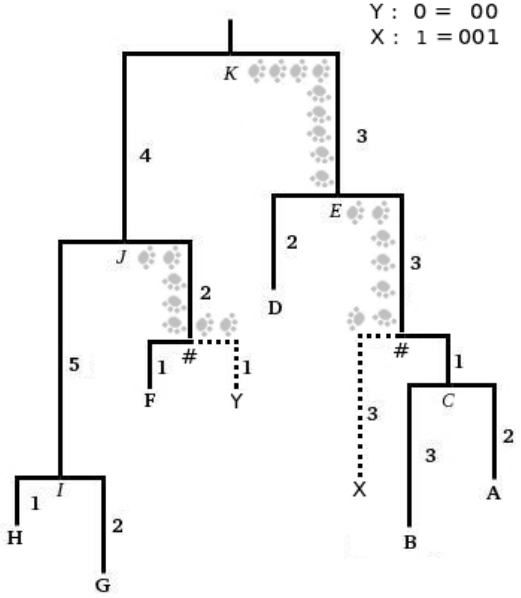
adicionar 2 espécies

Y : 1 = 01
X : 2 = 0010



adicionar 2 espécies

Y : 0 = 00
X : 1 = 001



Algoritmo

```
1  for todas as threads do in parallel
2      faça uma copia da árvore original para area de acesso exclusivo
3      while houver espécies a serem inseridas do
4          definir, randomicamente, a posição para inserir a nova espécie
5          inserir a nova espécie e o novo ancestral
6          if irmã da nova espécie também é uma espécie then
7              definir o comprimento igual ao da sua espécie irmã
8          else
9              examinar o clado irmão até chegar a uma de suas espécies
10             utilizar a distância acumulada para definir o comprimento do
               ramo
11         end
12     end
13 end
```

Estrutura de dados atualizada com 1 (uma) espécie

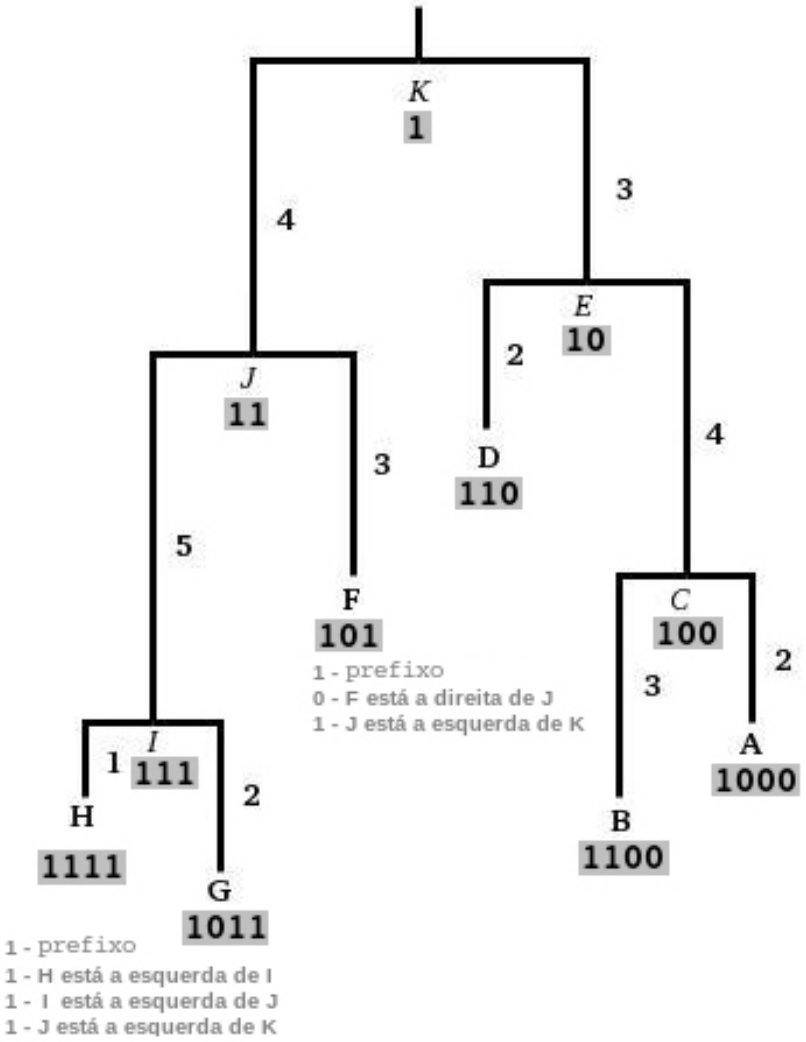
	espécies iniciais						novas espécies			novos ancestrais		ancestrais iniciais				
indice	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*	*13*	*14*	*15*
espécie	**A**	**B**	**D**	**G**	**H**	**F**	***X***	*Y*	-	*?*	*#*	**I**	**J**	**C**	**E**	**K**
pai	13	13	14	*10*	11	12	*10*	*12*	-	*?*	*11*	12	15	14	15	-1
filho esquerda	-2	-2	-2	-2	-2	-2	*-2*	*-2*	-	*?*	*6*	3	11	0	13	14
filho direita	-2	-2	-2	-2	-2	-2	*-2*	*-2*	-	*?*	*3*	4	5	1	2	12
comprimento do ramo	2	3	2	*1*	1	3	*1*	*?*	-	*?*	*1*	5	4	4	3	0
característica (traits)	3,3	2,4	6,0	3,4	5,6	2,1	*4,3*	*5,5*	-	*0,0*	*0,0*	0,0	0,0	0,0	0,0	0,0

Visão geral

Método para cálculo das distâncias

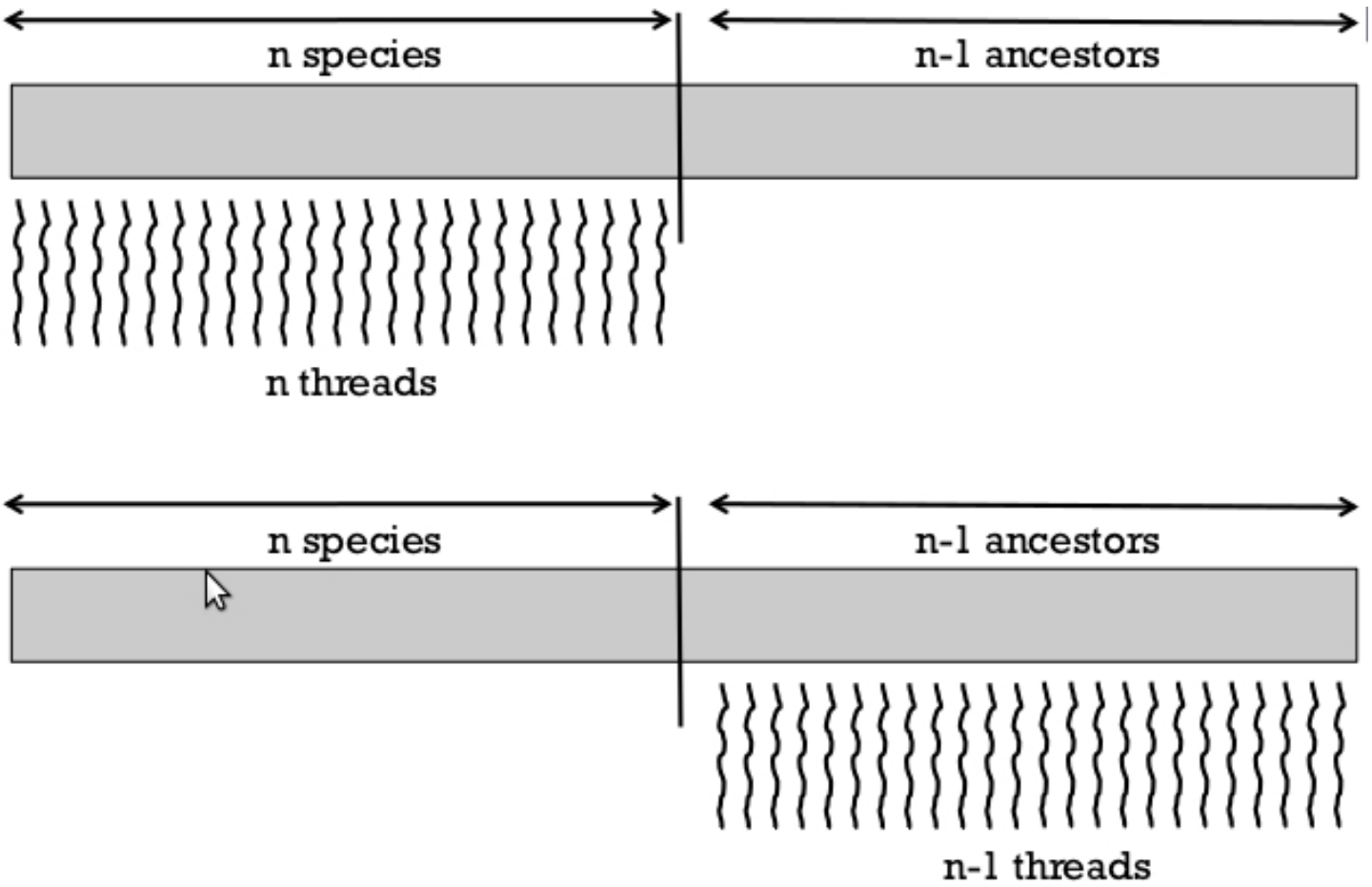
- É uma matriz quadrada, que relaciona todas as espécies
- Calcula todas as distância entre cada espécies da árvore
- A distância entre uma espécie e outra corresponde a soma dos comprimentos dos ramos que as interligam
- É uma matriz simétrica
- A diagonal principal é trivial, com todos valores iguais a zero
- A soma realizada através do *caminhamento* pelos ramos que interligam duas espécies, é redundante.
- Nossa solução realiza a soma para cada espécie até a raiz e armazena os resultados em uma *tabela hash*. Em seguida utilizamos esses dados para calcular as distâncias.

Tabela *hash*



Nó	Chave	Distância
K	1	0
J	11	4
E	10	3
I	111	9
F	101	7
D	110	5
C	100	7
H	1111	10
G	1011	11
B	1100	10
A	1000	9

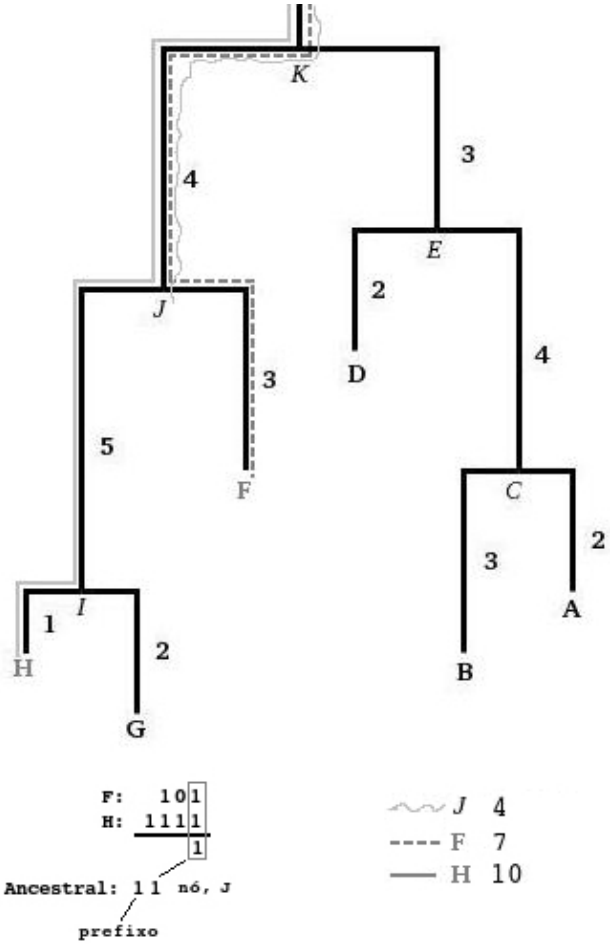
Calculo das distâncias até a raiz



Algoritmo

```
1  for todas as threads do in parallel
2      associar a thread com uma espécie ou ancestral
3      while nó pai não for a raiz do
4          acumular a distância e o caminho percorrido (na forma binária)
5          seguir para o nó pai
6      end
7      armazenar a distância e o caminho binário em uma tabela hash
8      sync_threads (barreira de sincronização )
9      associar uma thread com um elemento do vetor
10     for k ← 1 to (quantidade de espécies/ 2) do
11         associar um elemento do vetor para espécie da matriz
12         consultar na tabela hash a distância das espécies até a raiz
13         comparar os caminhos binários do par de espécies
14         encontrar o LCA e consultar na tabela hash a distância do mesmo
           até a raiz
15         calcular a distância entre o par de espécies
16         distância do 1º + distancia do 2º - 2 vezes a distância do LCA
17         armazenar a distância em um vetor que representa a matriz
18     end
19 end
```

LCA (lowest common ancestor)



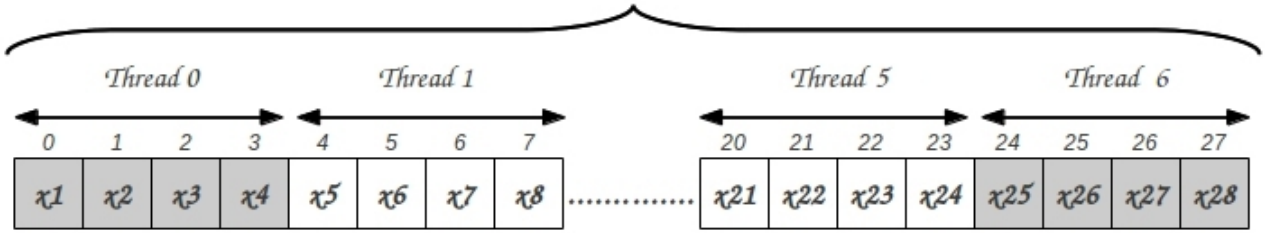
Qual a distância entre H e F?
$LCA(H, F) = J$

$dist(H, K) = 10$
$dist(F, K) = 7$
$dist(J, K) = 4$

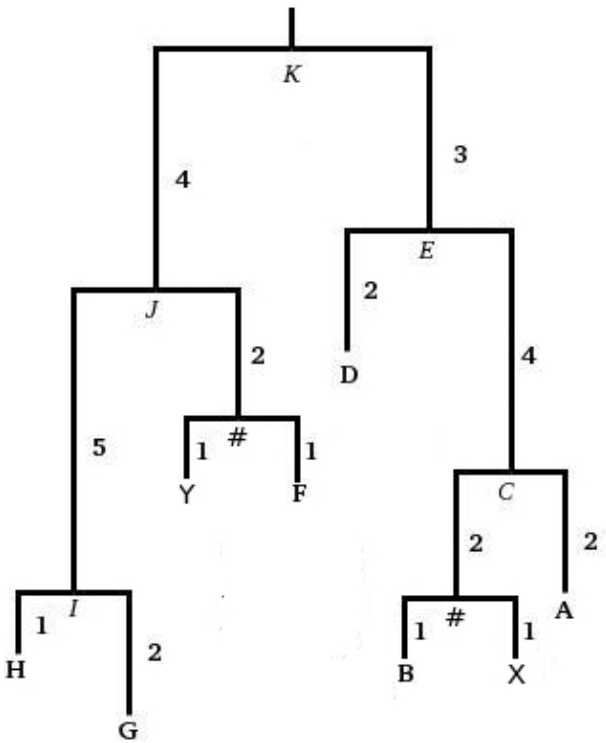
$dist(H, F) = dist(H, K) + dist(F, K) - 2 * dist(J, K)$
$dist(H, F) = 10 + 7 - 2 * 4 = 9$

Divisão da carga de trabalho de forma igualitária

	0	1	2	3	4	5	6	7
0	0	x1	x2	x3	x4	x5	x6	x7
1	x1	0	x8	x9	x10	x11	x12	x13
2	x2	x8	0	x14	x15	x16	x17	x18
3	x3	x9	x14	0	x19	x20	x21	x22
4	x4	x10	x15	x19	0	x23	x24	x25
5	x5	x11	x16	x20	x23	0	x26	x27
6	x6	x12	x17	x21	x24	x26	0	x28
7	x7	x13	x18	x22	x25	x27	x28	0



Árvores e sua matriz de distância



	A	X	B	D	F	Y	G	H
A	-	5	5	8	16	16	20	19
X		-	2	9	17	17	21	20
B			-	9	17	17	21	20
D				-	12	12	16	15
F					-	2	10	9
Y						-	10	9
G							-	3
H								-

Fórmula

$$I = (\frac{n}{S}) \left[\frac{\sum(\sum(W_{ij}(y_i - \overline{y}) \cdot (y_j - \overline{y}))}{\sum(y_i - \overline{y})^2} \right] \quad (1)$$

Onde:

- n: é o número de espécies
- y: representa a variável analisada (representa uma característica ou *traits*)
- $\overline{y}$: é a média de y
- W_{ij}: é o valor 1 ou zero, representa a conectividade. Indica se o par de distâncias da matriz simétrica esta ou não dentro da classe.
- S: é a soma dos elementos de conectividades em cada classe de distância

I de Moran por classe de distância

	A	X	B	D	F	Y	G	H
A	-	5	5	8	16	16	20	19
X		-	2	9	17	17	21	20
B			-	9	17	17	21	20
D				-	12	12	16	15
F					-	2	10	9
Y						-	10	9
G							-	3
H								-

	Faixa	Grupo de espécies por faixa
Classe 1	2,00 – 8,33	(X,A) (B,A) (D,A) (B,X) (Y,F) (H,G)
Classe 2	8,34 – 14,67	(D,X) (D,B) (F,D) (Y,D) (G,F) (H,F) (G,Y) (H,Y)
Classe 3	14,68 – 21,00	(F,A) (Y,A) (G,A) (H,A) (F,X) (Y,X) (G,X) (H,X) (F,B) (Y,B) (G,B) (H,B) (G,D) (H, D)

Algoritmo

```
1  armazenar o vetor com as classes de distância na memória compartilhada
2  criar e inicializar variáveis compartilhadas, utilizada nas somas parciais
3  calcular a media das características
4  for todas as threads do in parallel
5      for classe ← 1 to (número de classes) do
6          for k ← 1 to (quantidade de espécies / 2) do
7              if distância entre as espécies pertence a classe then
8                  associar o elemento do vetor de distância as espécies da
                   matriz
9                  calcular e acumular o produto entre a diferença das espécies
                   e a média
10             end
11         end
12         realiza, atomicamente, o somatório para calcular o I de Moran
13         sync_threads (barreira de sincronização)
14         apenas uma thread calcula o I de Moran para a classe
15         inicializar variáveis compartilhadas
16         sync_threads (barreira de sincronização)
17     end
18 end
```

Sumário

Plataforma computacional

- Intel Core2 Duo 1.6GHz, 2GB RAM, NVIDIA Tesla C1060 e Sistema Operacional Linux (Ubuntu 11.04).
- A Tesla contem com conjunto de 30 Multiprocessadores de streaming (SMs), cada um com 8 núcleos de processamento (SPs) (total de 240 cores) com velocidade de clock igual 1.3GHz, 4 GB de memória global e 16 KB de memória compartilhada.

Programa e ferramentas

- Os algoritmos foram desenvolvidos em C/C++ e CUDA C/C++ 4.0.
- Comparamos nossos resultados com os resultados produzidos no programa Phylocom, um conhecido software de código aberto para analises filogenética. Apenas os resultados da matriz patrística foi comparado com este software, pois o mesmo não calcula as outras fases.
- Nós desenvolvemos uma versão serial de nosso algoritmo paralelo para efeito de comparação.

Conjunto de dados

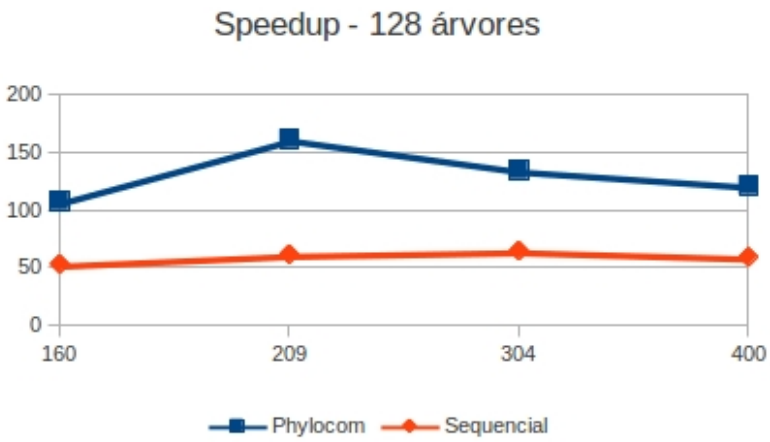
Os resultados foram experimentados sobre quatro filogenias:

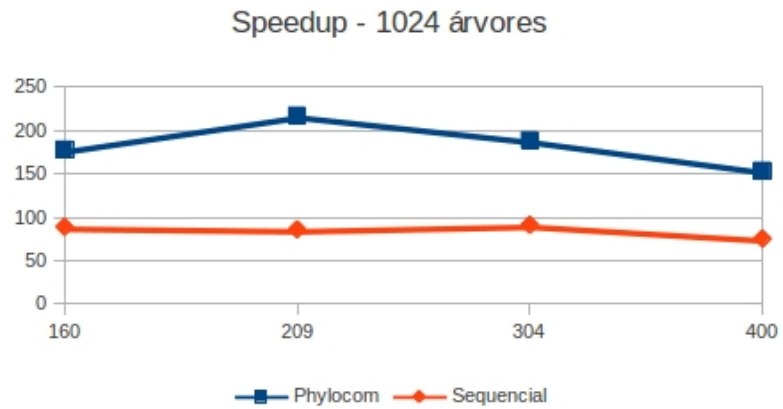
- **Carnivores:** Essa filogenia contém 209 espécies e não possui nenhuma espécie perdida/desconhecida. A característica (ou trait) disponível é o tamanho de corpo.
- **Hummingbirds:** são 304 espécies de beija-flores no total, sendo 158 PUT. O *trait* disponível é o tamanho do corpo.
- **DummyA:** espécies artificiais, em uma filogenia completamente balanceada. O trait foi criado artificialmente. São 160 espécies no total, sendo 32 delas PUT.
- **DummyB:** espécies artificiais, em uma filogenia completamente balanceada. O trait foi criado artificialmente. São 400 espécies no total, sendo 272 delas PUT.

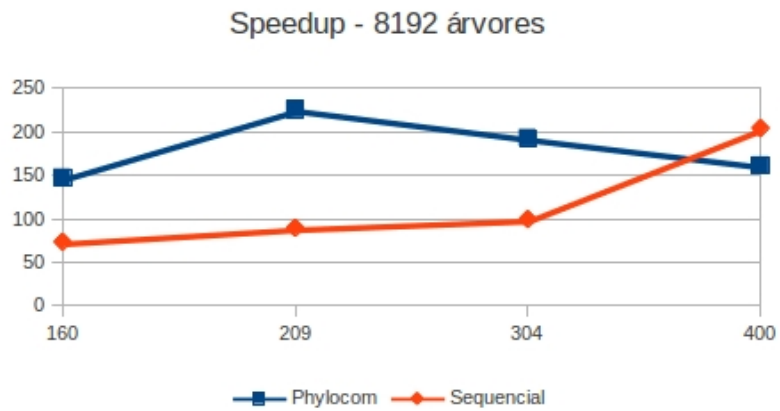
Matriz de distância - Tabela

	Espécies	Phylocom	Sequencial	Paralelo	
				Execução	Cópia
128 árvores	DummyA (160)	0,64 (0,05)	0,31 (0,01)	0,01	0,00
	Carnivores (209)	1,42 (0,06)	0,57 (0,04)	0,01	0,00
	Hummingbirds (304)	2,86 (0,07)	1,41 (0,09)	0,02	0,00
	DummyB (400)	4,63 (0,07)	2,25 (0,03)	0,04	0,00
1024 árvores	DummyA (160)	4,85 (0,01)	2,43 (0,02)	0,04	0,01
	Carnivores (209)	10,95 (0,03)	4,35 (0,01)	0,05	0,00
	Hummingbirds (304)	22,40 (0,07)	10,85 (0,61)	0,12	0,01
	DummyB (400)	36,86 (0,05)	18,09 (0,67)	0,25	0,01
8192 árvores	DummyA (160)	38,83 (0,02)	19,39 (0,03)	0,26	0,02
	Carnivores (209)	87,47 (0,06)	34,71 (0,10)	0,41	0,03
	Hummingbirds (304)	178,83 (0,07)	92,49 (2,79)	0,93	0,05
	DummyB (400)	294,60 (0,10)	372,83 (28,04)	1,82 (0,02)	0,07

Matriz de distância - Speedup







Todas as etapas - Tabela

128 árvores

Espécies	Sequêncial			Paralelo			
	Inserir	Matriz	I Moran	Inserir	Matriz	I Moran	Cópia
160	0,00	0,31(0,01)	0,34	0,01	0,01	0,01	0,00
304	0,01	1,41(0,08)	1,25(0,07)	0,02	0,02	0,02	0,00
400	0,02	2,25(0,03)	2,25(0,06)	0,03	0,04	0,06	0,00

1024 árvores

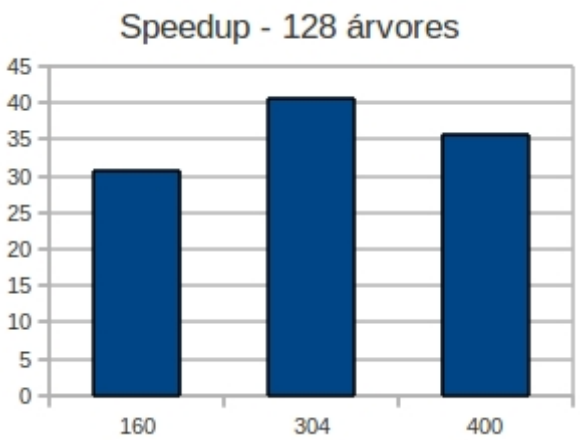
Espécies	Sequêncial			Paralelo			
	Inserir	Matriz	I Moran	Inserir	Matriz	I Moran	Cópia
160	0,02	2,43(0,02)	2,70(0,01)	0,14	0,04	0,04	0,01
304	0,08	10,85(0,61)	10,08(0,82)	0,17	0,12	0,14	0,01
400	0,12	18,09(0,67)	18,03(1,11)	0,20	0,24	0,31(0,08)	0,01

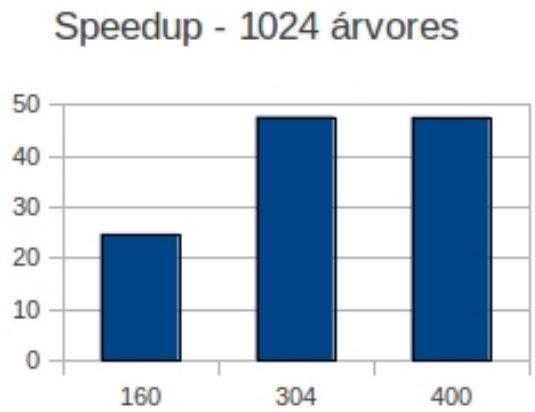
8192 árvores

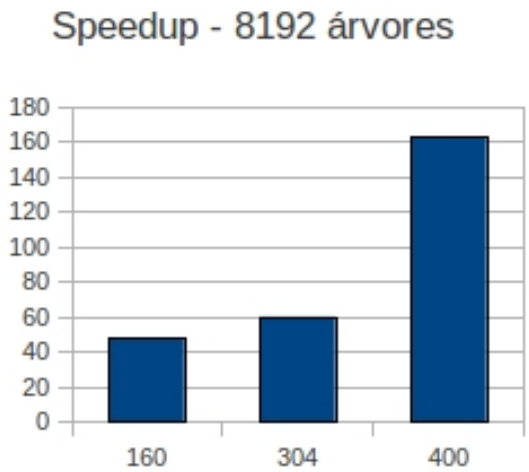
Espécies	Sequêncial			Paralelo			
	Inserir	Matriz	I Moran	Inserir	Matriz	I Moran	Cópia
160	0,14	19,39	21,58	0,33	0,26	0,39	0,02
304	0,66	92,49	78,90	0,78	0,93	1,08	0,05
400	0,95	372,83	305,60	1,02	1,82	2,18	0,07

Espécies	Sequêncial			Paralelo			
	Inserir	Matriz	I Moran	Inserir	Matriz	I Moran	Cópia
160	(0,00)	(0,03)	(0,02)	(0,00)	(0,00)	(0,01)	(0,00)
304	(0,04)	(2,79)	(4,67)	(0,00)	(0,00)	(0,04)	(0,00)
400	(0,07)	(28,04)	(33,86)	(0,00)	(0,02)	(0,02)	(0,00)

Todas as etapas - Speedup







Sumário

- Estrutura de dados: exploramos o uso de estruturas de dados eficientes, que permitem acesso aglutinado a memória.
- Obtivemos *speedups* muito bons nas fases: calculo da matriz de distância e calculo do Índice de Moran.
- Conseguimos atingir *speedups* máximo de 220x, quando comparamos com o programa phylocom.
- Atingimos *speedups* máximo de 60x, ao executar todas etapas e compararmos com a versão sequencial.

- No cálculo da matriz de distância alcançamos um nível de paralelismo extremamente alto, chegando a lançar milhões de threads. Esse, foi o caso da filogenia dos beija-flores (hummingbirds), que possui 304 espécies. Ao simularmos 8.192 árvores, conseguimos lançar um grid com 377.290.752 (mais de 377 milhões) de threads.
- Índice de Moran: utilizamos acesso a memória compartilhada para armazenar um pequeno conjunto de dados: classes de distância e variáveis acumuladoras.

- Além disso, pretendemos utilizar diversos outros métodos estatísticos que requerem a matriz de distância patrística como entrada.
- Implementar um sistema completo de simulação (utilizando o método de Monte Carlo) para estudos comparativos filogenéticos, incluindo o cálculo de diferente coeficientes (além do I de Moran) e análises estatísticas.

Dúvidas?