Estudo Comparativo de Técnicas de Escalonamento de Tarefas Dependentes para Grades Computacionais

Candidato
Alvaro Henry Mamani Aliaga[1]

Orientador
Alfredo Goldman

Instituto de Matemática e Estatística
Departamento de Ciência da Computação
Universidade de São Paulo

alvaroma@ime.usp.br

22 de Agosto de 2011

[1]O aluno recebeu apoio financeiro do CNPq, processo 133147/2009-6

Roteiro

Introdução

- Necessidade de poder computacional: mineração de dados, previsão do tempo, processamento de imagens médicas, ...
- Aumento na disponibilidade de computadores poderosos e na interligação de redes de alta velocidade
- Computação em grade

  Uma alternativa para obter grande capacidade processamento
- Escalonamento de tarefas consiste em alocar tarefas de uma aplicação em recursos computacionais, com o intuito de minimizar o *Makespan*

Escalonadores

- **Escalonadores**
  - OAR
  - Condor
  - Torque
- **Middlewares**
  - Boinc
  - InteGrade
  - OurGrid
  - XtremWeb

Escalonadores

- **Escalonadores**
  - OAR
  - Condor
  - Torque
- **Middlewares**
  - Boinc
  - InteGrade
  - OurGrid

    Algoritmos de Escalonamento
    - ★ *Workqueue*
    - ★ *Workqueue with Replication*
    - ★ *Storage Affinity*
  - XtremWeb

Motivação

- Necessidade de grande capacidade de processamento
- Uso correto da capacidade do processamento
- **Escalonamento** é um grande desafio pelas características da grade
- Várias abordagens de escalonamento propostas
- O escalonamento em *middlewares* geralmente usa políticas de escalonamento básicas

Objetivos

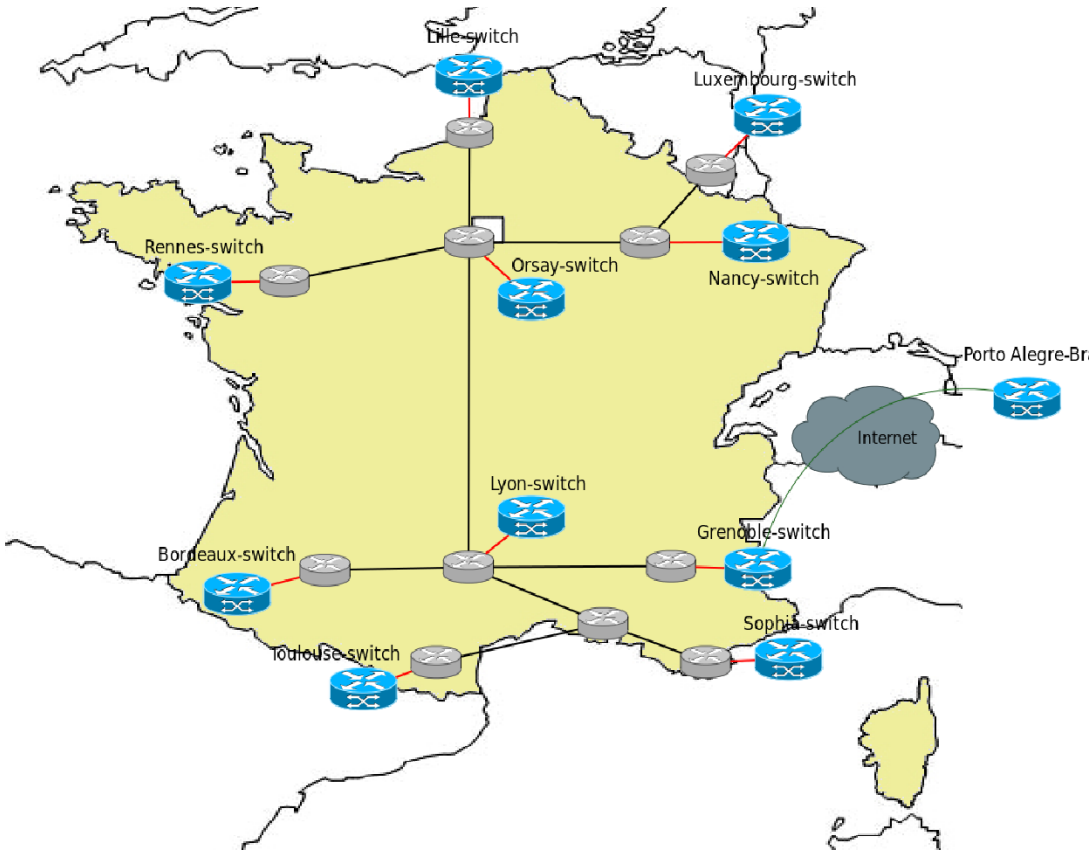
- **Objetivo geral**
  - Comparar técnicas de escalonamento para grades computacionais sobre diferentes cenários
- **Objetivos específicos**
  - Propor uma metodologia que baseada em características tanto das aplicações quanto das arquiteturas da grade seja possível decidir qual algoritmo oferece melhor desempenho
  - Determinar se para um dado tipo de aplicação é possível uma comparação usando escalabilidade

DAS-3

- *Distributed ASCI Supercomputer 3*
- Arquitetura composta por cinco aglomerados heterogêneos geograficamente distribuídos pela Holanda
- Possui 272 processadores

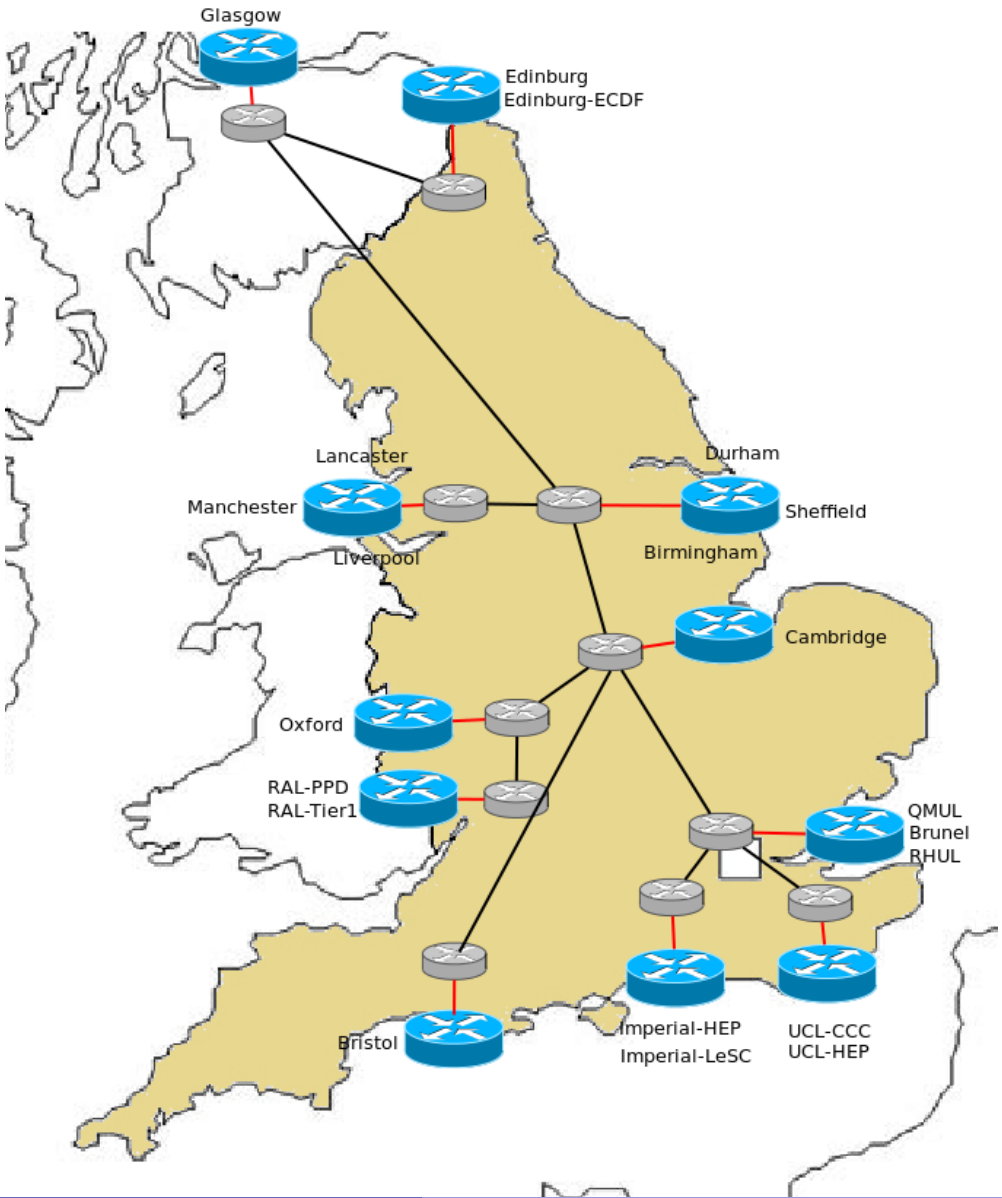
Grid5000

- Arquitetura científica criada para o estudo de sistemas paralelos e distribuídos de larga escala espalhados pelo território francês
- Possui mais de 5000 processadores
- Nos experimentos são usados 462 processadores, agrupados em doze aglomerados



GridPP

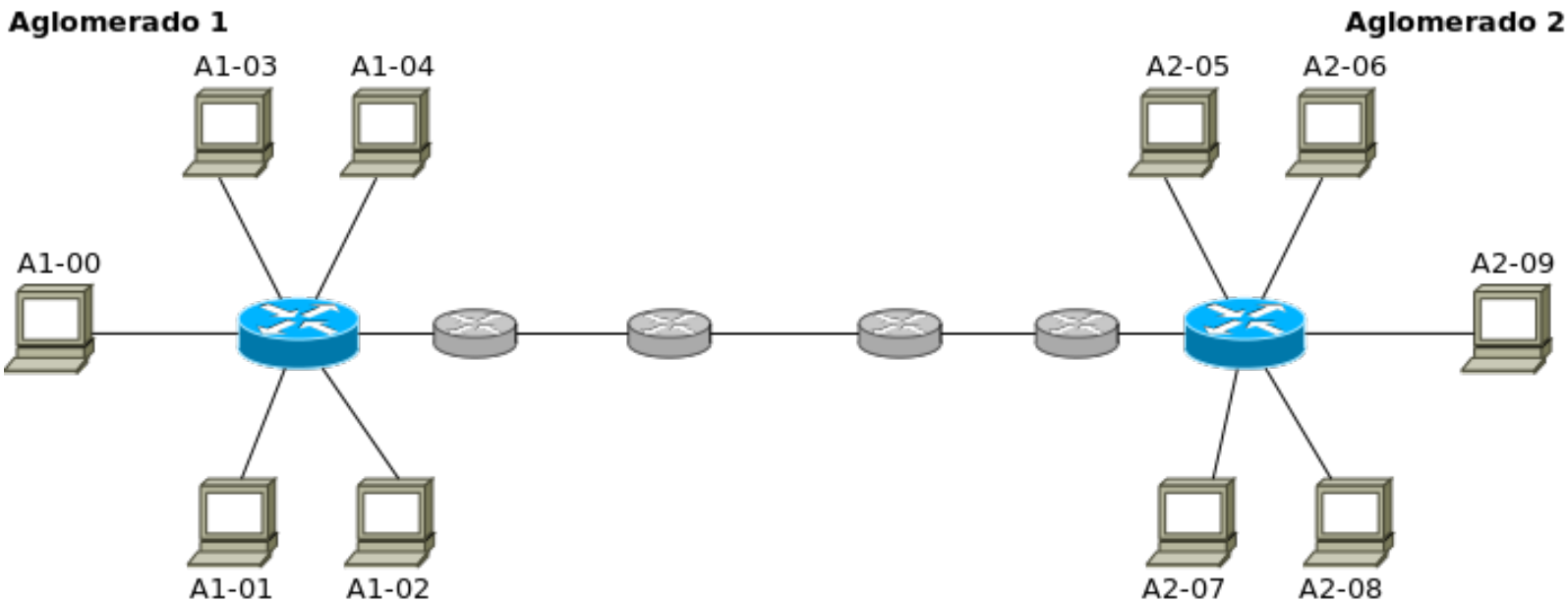
- É uma arquitetura colaborativa entre físicos e cientistas da computação de 19 universidades do Reino Unido, o laboratório Rutherford e o CERN
- Possui mais de 7948 processadores
- Nos experimentos são usados 900 processadores, agrupados em treze aglomerados



SmallGrid

Características da Arquitetura

- Foram especificados dois aglomerados, com duas instâncias: homogênea e heterogênea



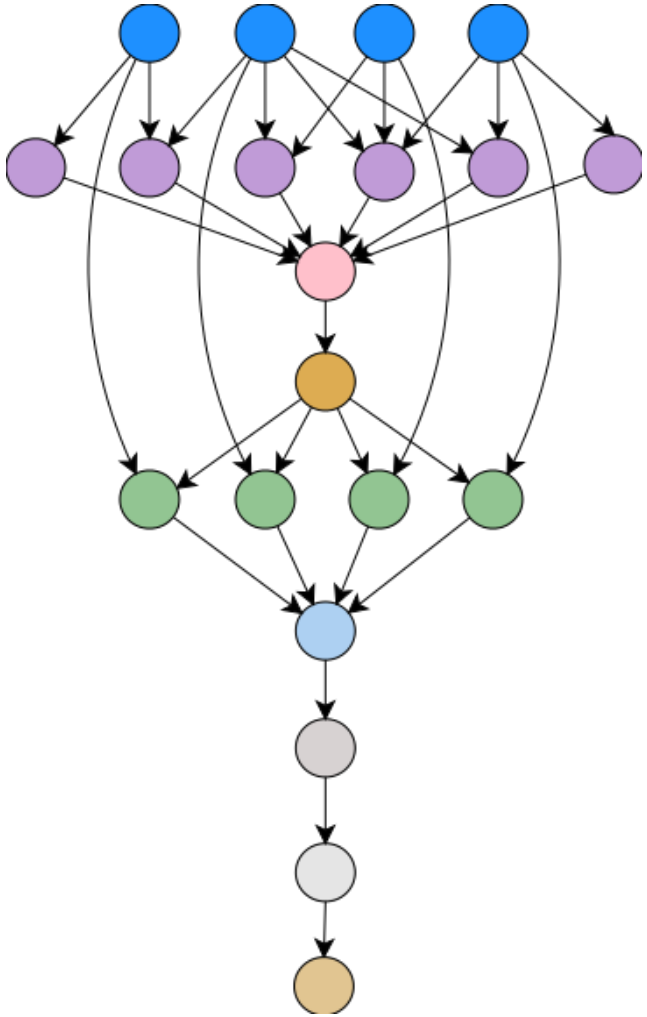
SmallGrid

Id	Poder Computacional (GFlops)	
	Homogêneo	Heterogêneo
A1-00	5,00	1,00
A1-01	5,00	2,00
A1-02	5,00	3,00
A1-03	5,00	4,00
A1-04	5,00	5,00
A2-05	5,00	5,00
A2-06	5,00	6,00
A2-07	5,00	7,00
A2-08	5,00	8,00
A2-09	5,00	9,00

Aplicações

Montage

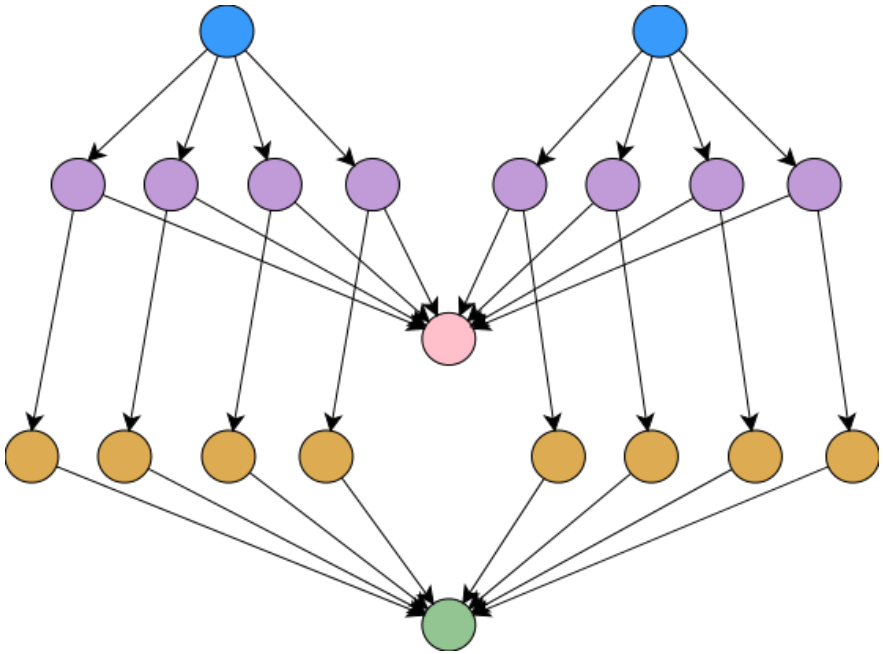
- É usada para gerar mosaicos personalizados do céu usando pontos de múltiplas imagens de entrada



Aplicações

CyberShake

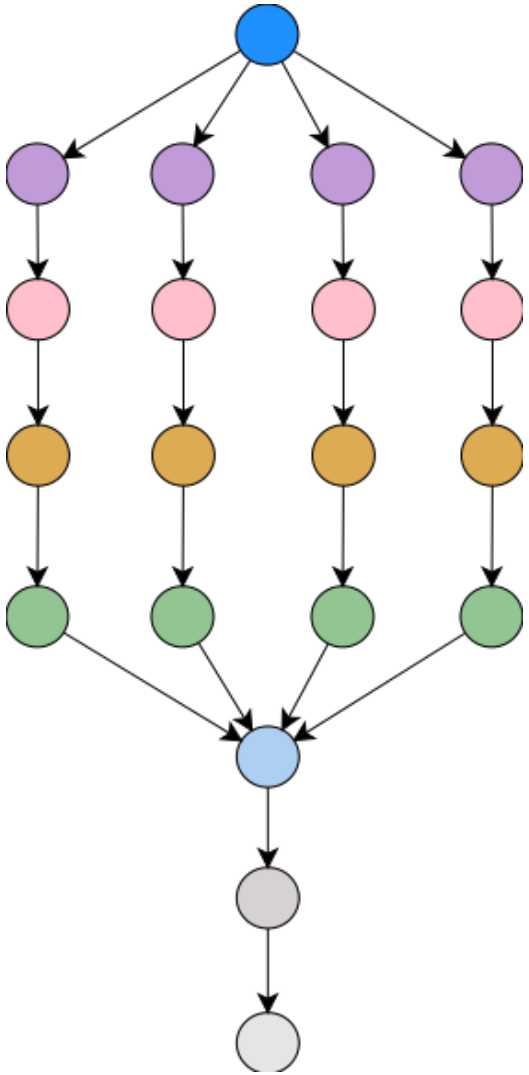
- O projeto tem como propósito calcular e analisar os riscos de terremoto usando técnicas de análise probabilística de risco sísmico



Aplicações

Epigenomics

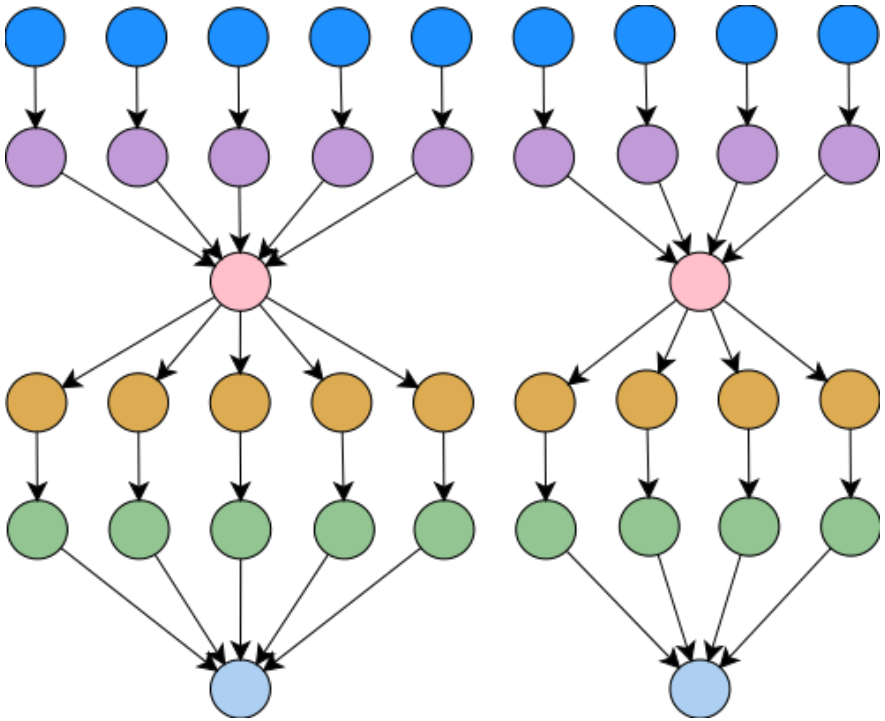
- É usada no mapeamento do estado epigenético de células humanas sobre uma grande escala genômica



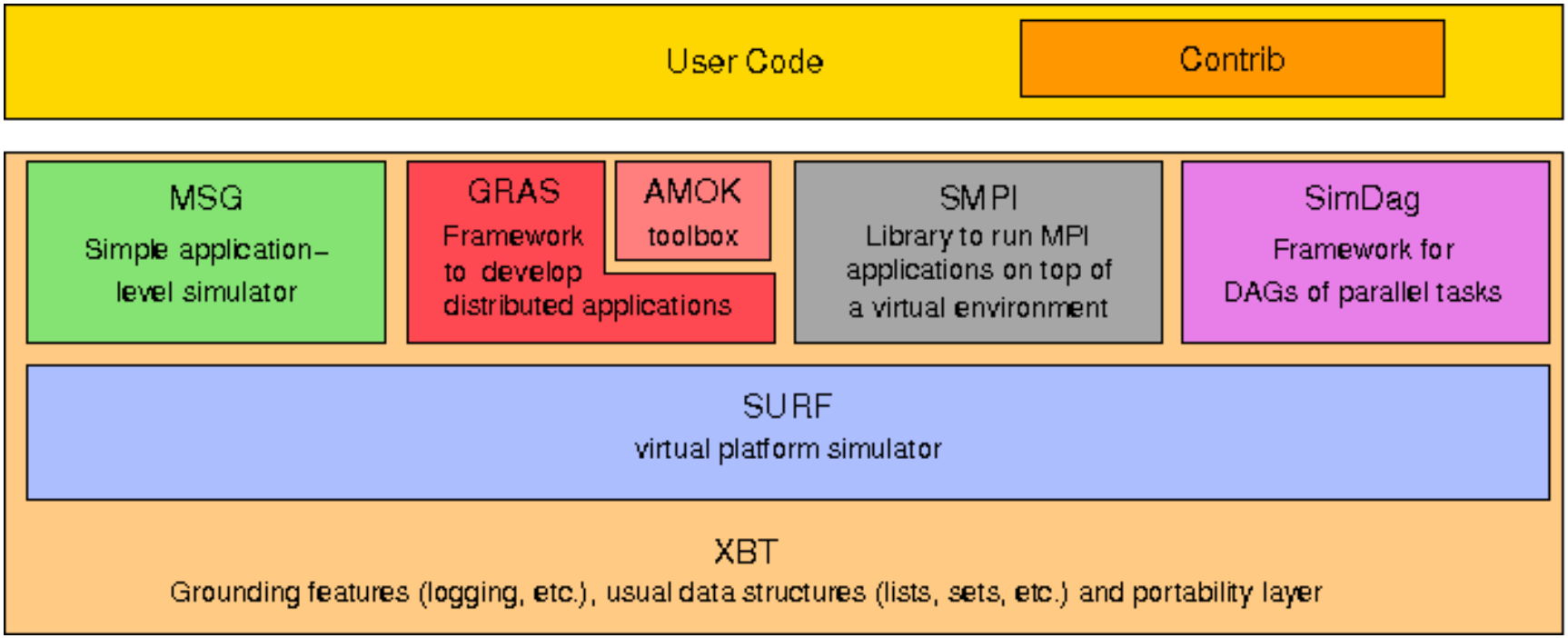
Aplicações

Ligo

- É usada para detectar ondas gravitacionais produzidas por vários eventos no universo



Simulador SimGrid



Casanova, Henri and Legrand, Arnaud and Quinson, Martin, SimGrid: a Generic Framework for Large-Scale Distributed Experiments, IEEE Computer Society, 2008.

HEFT, *Heterogeneous Earliest Finish Time*

Priorização de tarefas

- Atribuir prioridade às tarefas
- Calculo da prioridade, baseado na média dos custos de computação e custos de comunicação
- Lista de tarefas

Seleção de recursos

- Selecionar a tarefa t_i da lista com maior prioridade
- Para cada recurso $r \in R$ é calculado o *EST* e *EFT* de cada tarefa t_i
- r_j é alocada ao recurso que minimiza o EFT da tarefa t_i

Topcuouglu, Haluk et Al., Performance-Effective and Low-Complexity Task Scheduling for Heterogeneous Computing, IEEE Trans. Parallel Distrib. Syst., 2002.

CPOP, *Critical Path On a Processor*

Priorização de tarefas

- Atribuir prioridade às tarefas
- Calculo das prioridades baseados no custo de computação e comunicação
- $|CP|$ é o caminho crítico

Seleção de recursos

- *PCP* (*critical-path processor*)
- Se a tarefa selecionada está no caminho crítico, então é escalonada no recurso de caminho crítico
- Ela é atribuída a um recurso que minimiza o EFT

Topcuouglu, Haluk et Al., Performance-Effective and Low-Complexity Task Scheduling for Heterogeneous Computing, IEEE Trans. Parallel Distrib. Syst., 2002.

PCH, *Path Clustering Heuristic*

Seleção de tarefas e agrupamento

- Seleciona tarefas que formarão cada *cluster* que serão escalonadas no mesmo recurso
- A primeira tarefa que compõe um *cluster* cls_k é a tarefa não escalonada com maior prioridade

Seleção de recursos

- A seleção de recursos se dá através do cálculo de valores
- Qual recurso terminará a execução do *cluster* em menor tempo
- O fator que determina em qual recurso um *cluster* será escalonado é o *EST* do sucessor da último tarefa do *cluster* considerado

Bittencourt, Luiz F et Al., Uma Heurística de Agrupamento de Caminhos para Escalonamento de Tarefas em Grades Computacionais, SBRC, 2006.

Análise das Aplicações

- Distinguimos dois tipos: Aplicação Regular e Aplicação Irregular

Análise das Aplicações

- Distinguimos dois tipos: Aplicação Regular e Aplicação Irregular
- Por cada aplicação temos um conjunto de "traços de execução" (*traces*)
- A soma dos tempos de execução das tarefas (w_i) de um traço é denominada "carga do trabalho" (*workload*)

Análise das Aplicações

- Distinguimos dois tipos: Aplicação Regular e Aplicação Irregular
- Por cada aplicação temos um conjunto de "traços de execução" (*traces*)
- A soma dos tempos de execução das tarefas (w_i) de um traço é denominada "carga do trabalho" (*workload*)
- A carga de trabalho $W(T_A)$ de um traço de execução de uma aplicação T_A de tamanho n é dado por:

$$W(T_A) = \sum_{i=1}^{n} w_i$$

Análise das Aplicações

- Dada uma aplicação A, se $T_{A,n}$ é o conjunto de traços de tamanho n, a média da carga do trabalho $W(A, n)$ de uma aplicação de cada instância de tamanho n é dado por:

$$W(A, n) = \frac{1}{|T_{A,n}|} \sum_{T \in T_{A,n}} W(T)$$

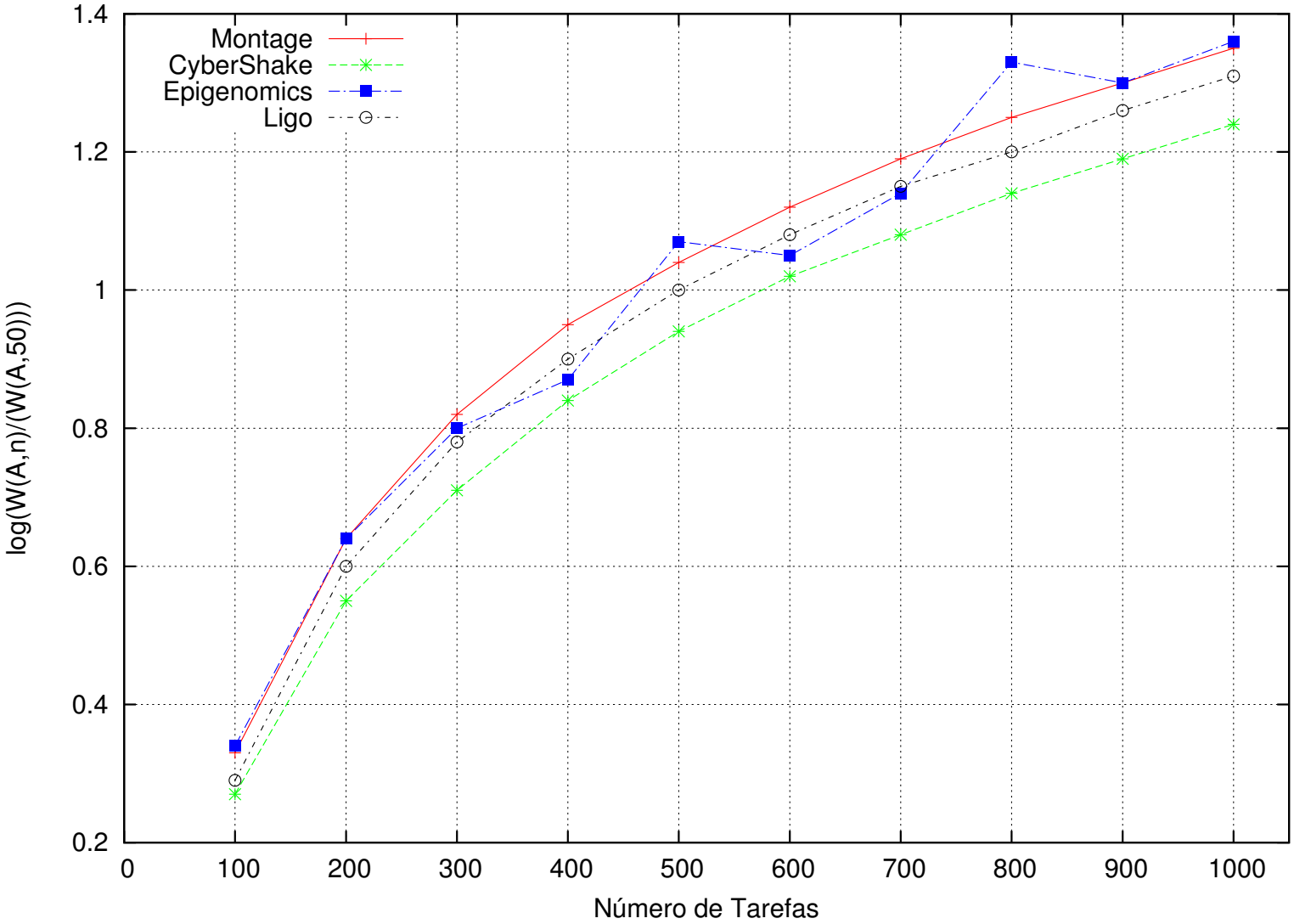
Análise das Aplicações

- Dada uma aplicação A, se $T_{A,n}$ é o conjunto de traços de tamanho n, a média da carga do trabalho $W(A, n)$ de uma aplicação de cada instância de tamanho n é dado por:

$$W(A, n) = \frac{1}{|T_{A,n}|} \sum_{T \in T_{A,n}} W(T)$$

- Dada uma aplicação A, chamamos a aplicação de irregular se $\exists n, m$ com $n < m$ tal que $W(A, n) > W(A, m)$

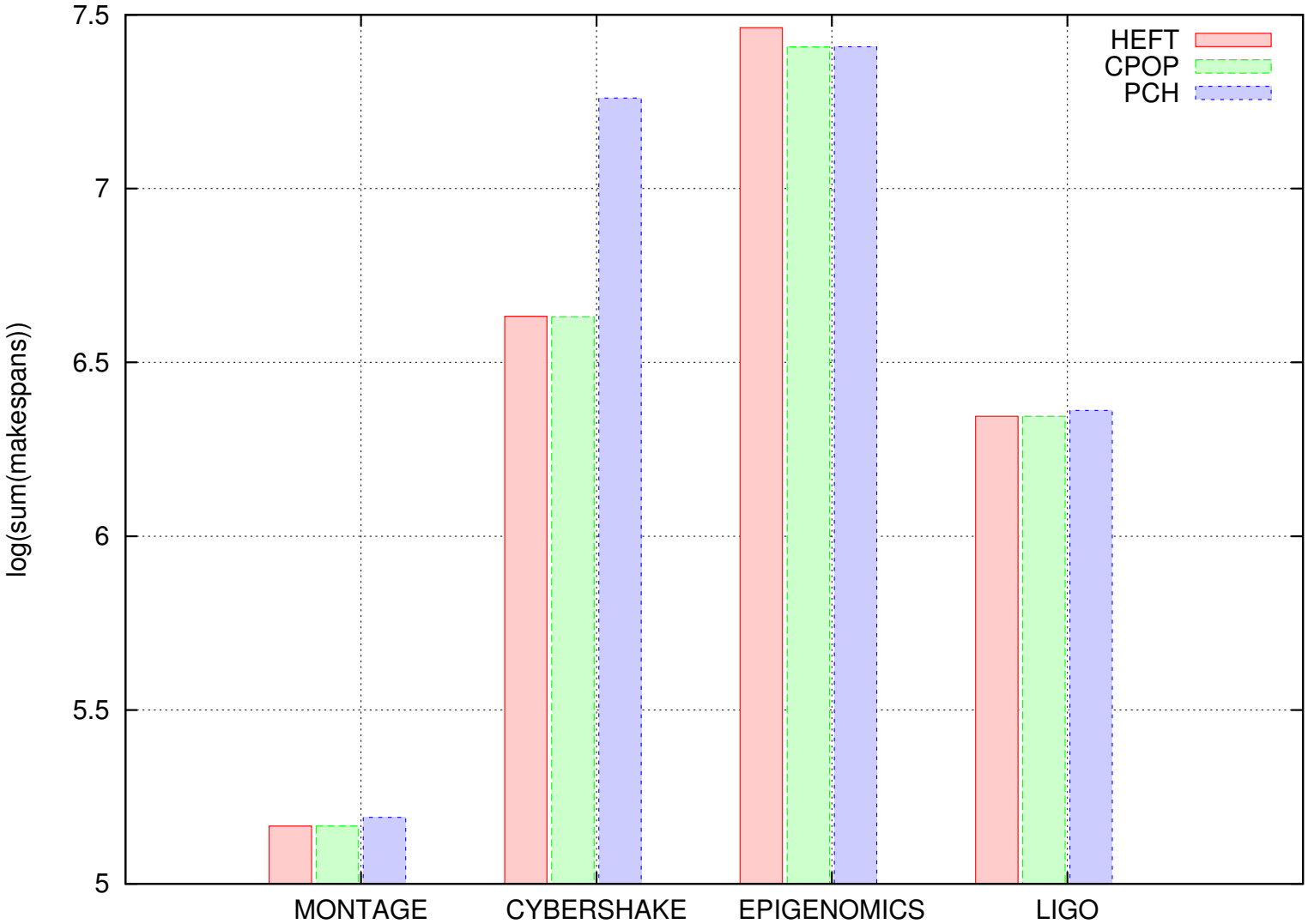
Análise das Aplicações



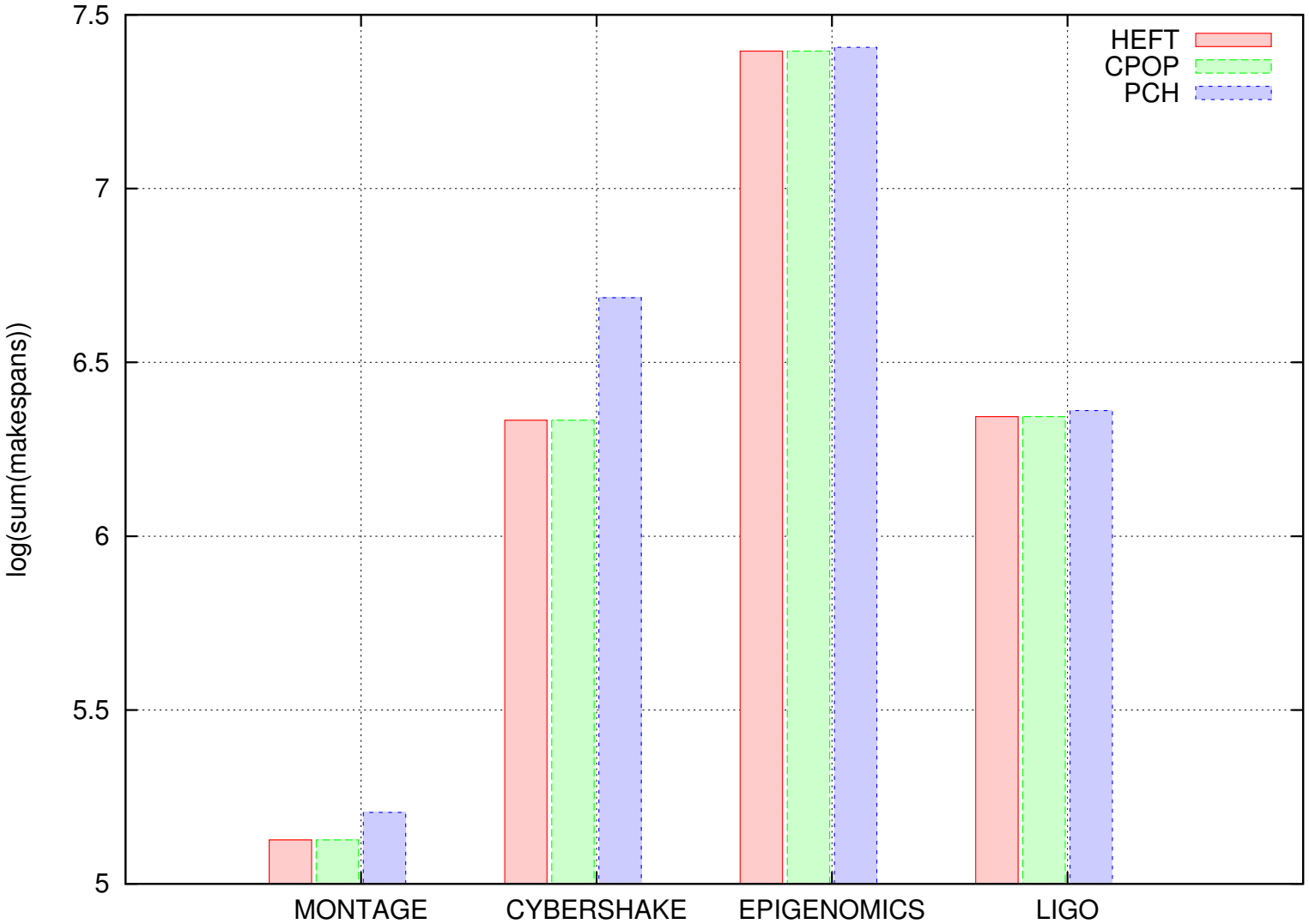
Principais Questões que Direcionam aos Experimentos

Critério	**Métricas**	**Questões & Configurações**
Desempenho	A soma total dos *Makespans*	Uma noção geral do algoritmo com o melhor desempenho
Escalabilidade	Média do *Makespan* pelo número de tarefas e nós da grade	A avaliação é feita para aplicações regulares sobre todas as grades
Adaptabilidade	Taxa entre o total do *Makespan* por grade e por aplicação	O intuito é identificar quais algoritmos são mais adaptativos sobre diferentes arquiteturas
Distribuição da Carga do Trabalho	Número de tarefas por nós da grade e tempo necessário para a comunicação entre elas	O intuito é entender qual algoritmo é o melhor na distribuição da carga do trabalho

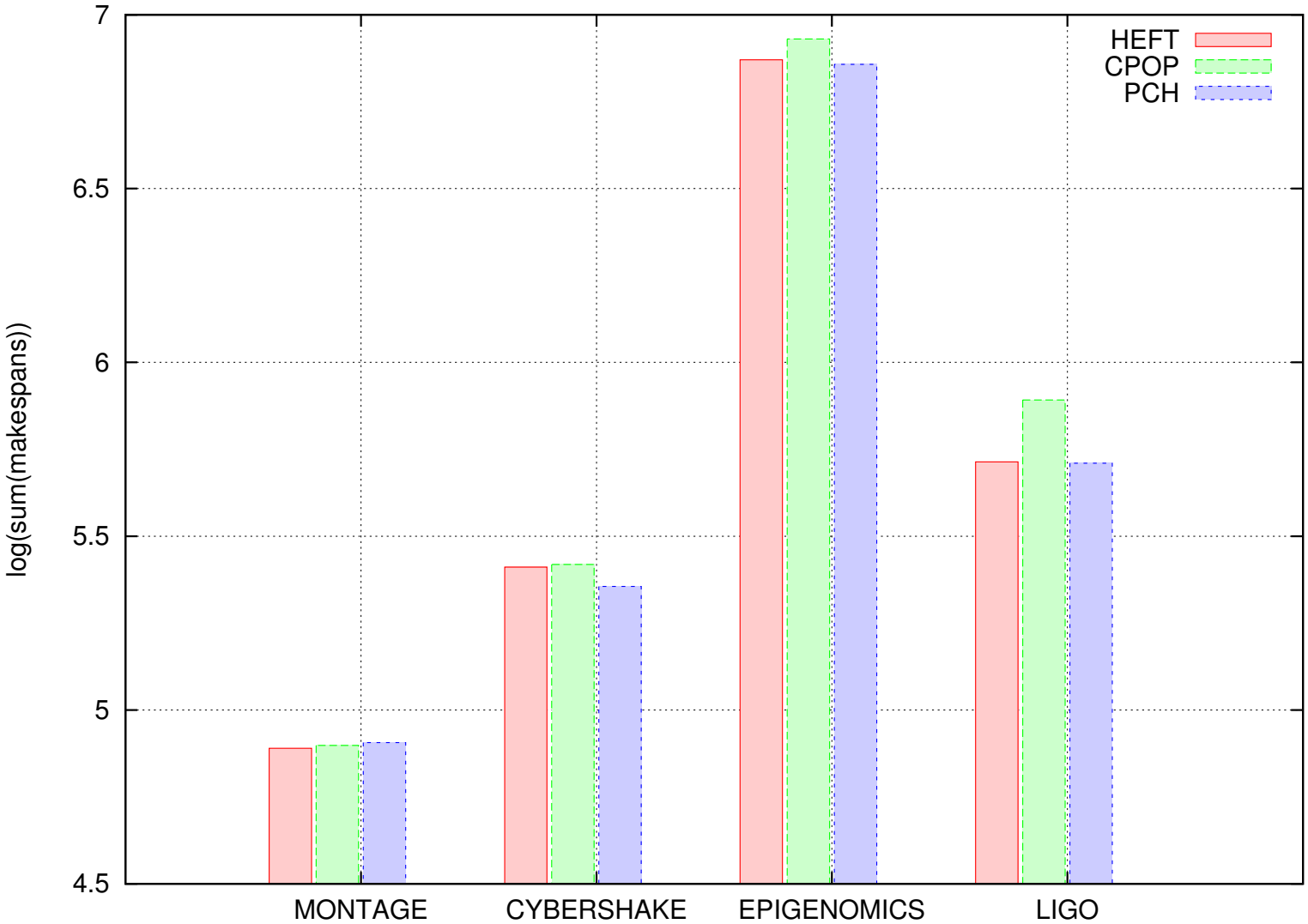
Desempenho - SmallGrid Homogênea



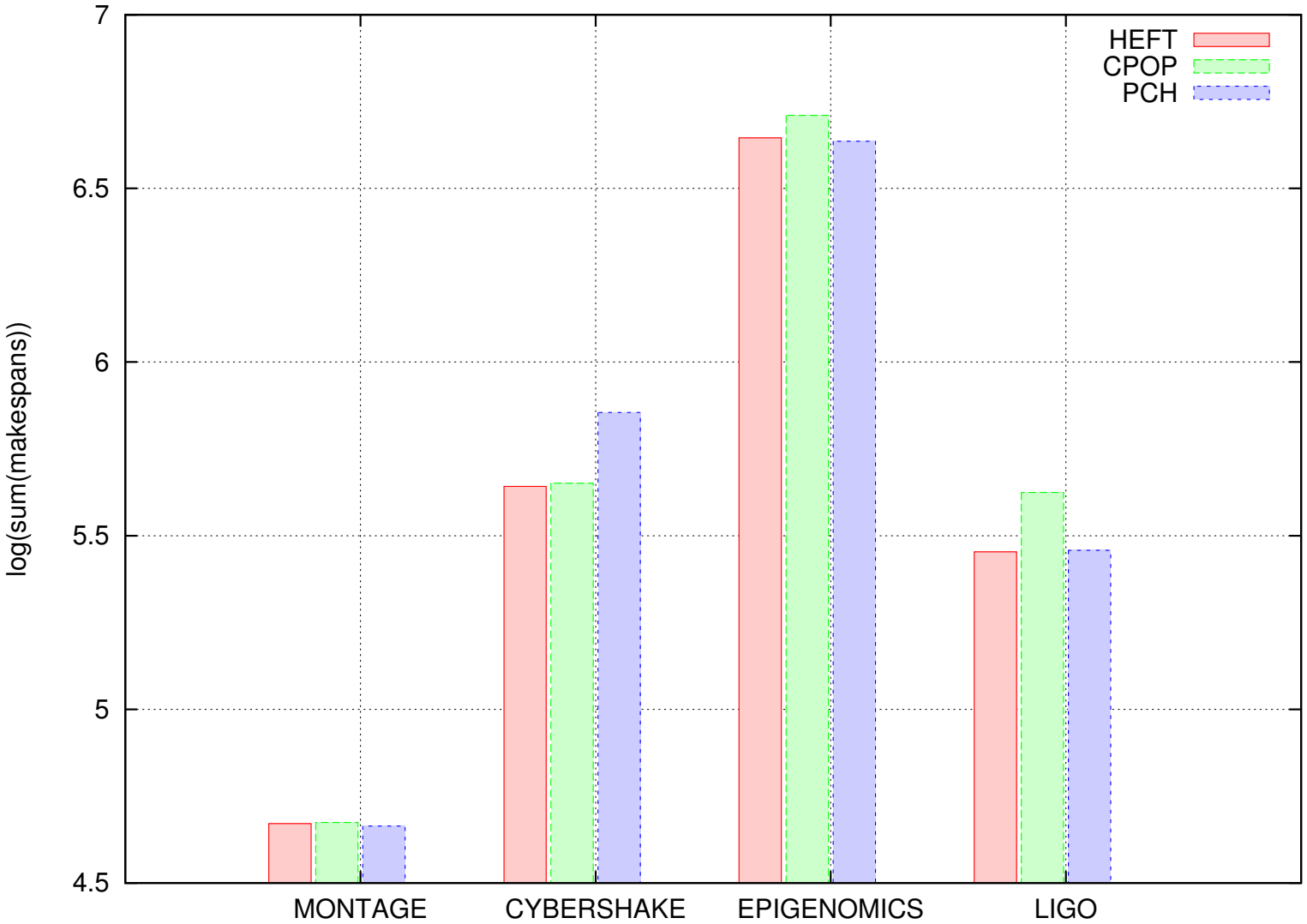
Desempenho - SmallGrid Heterogênea



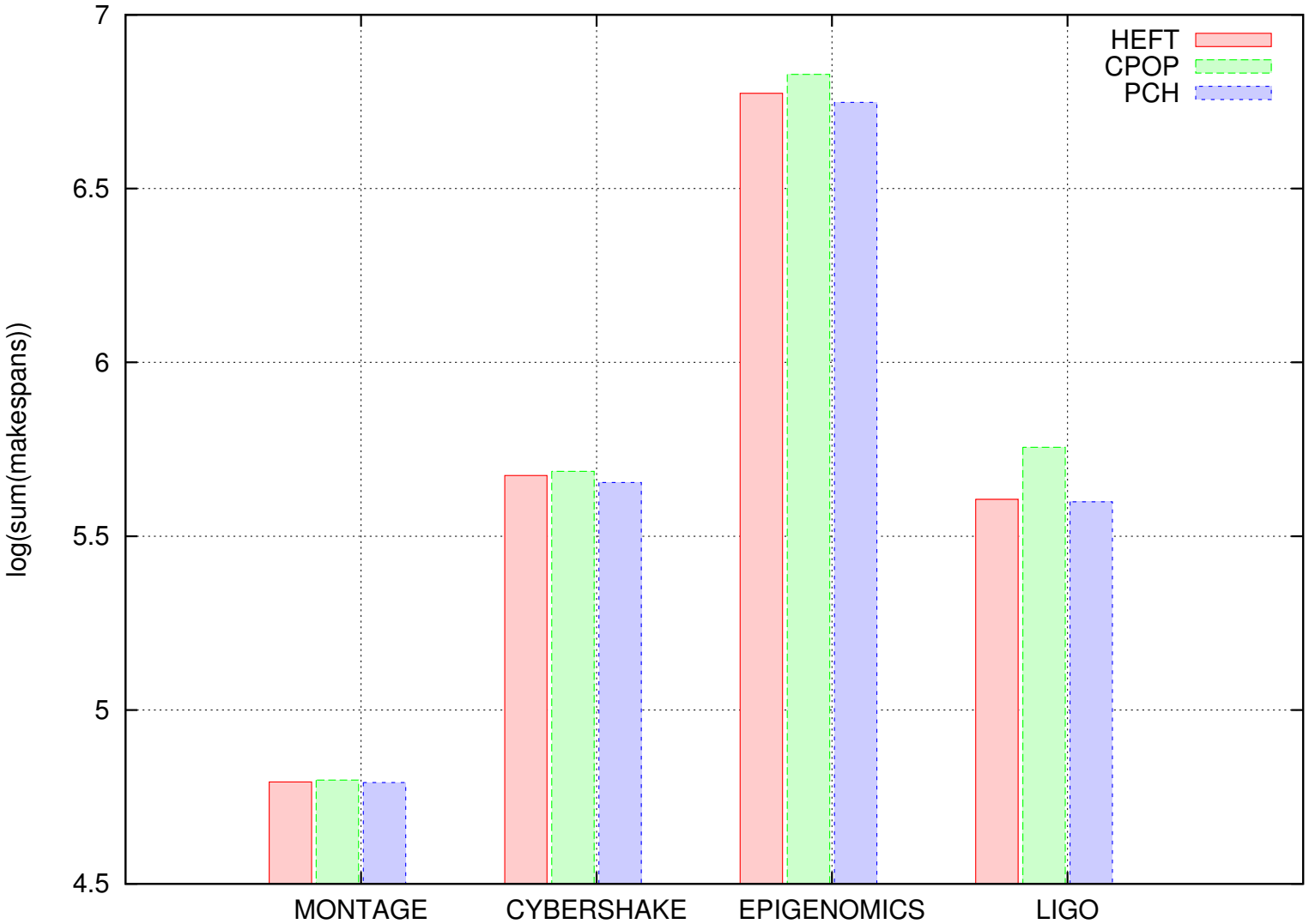
Desempenho - DAS-3



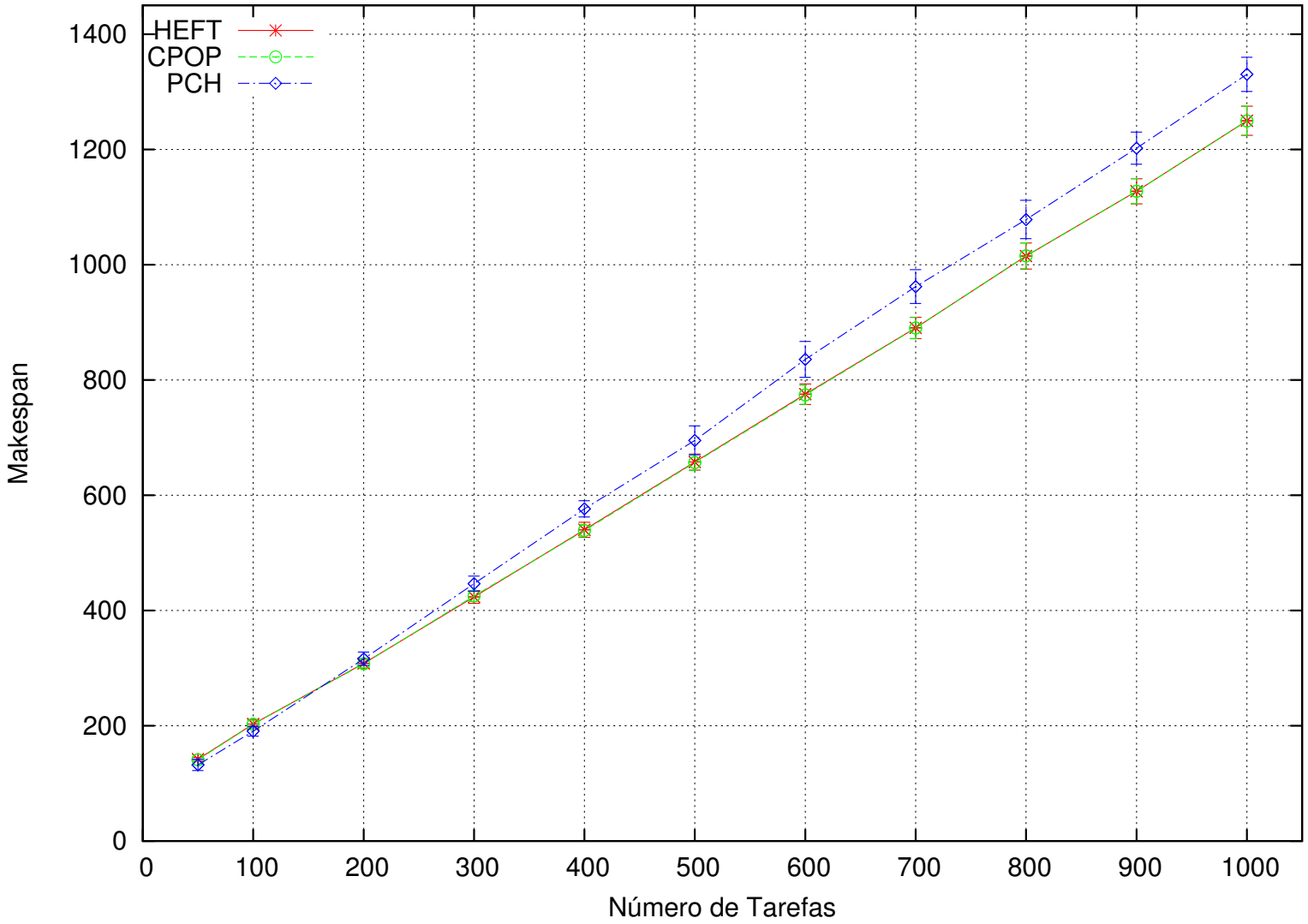
Desempenho - Grid5000



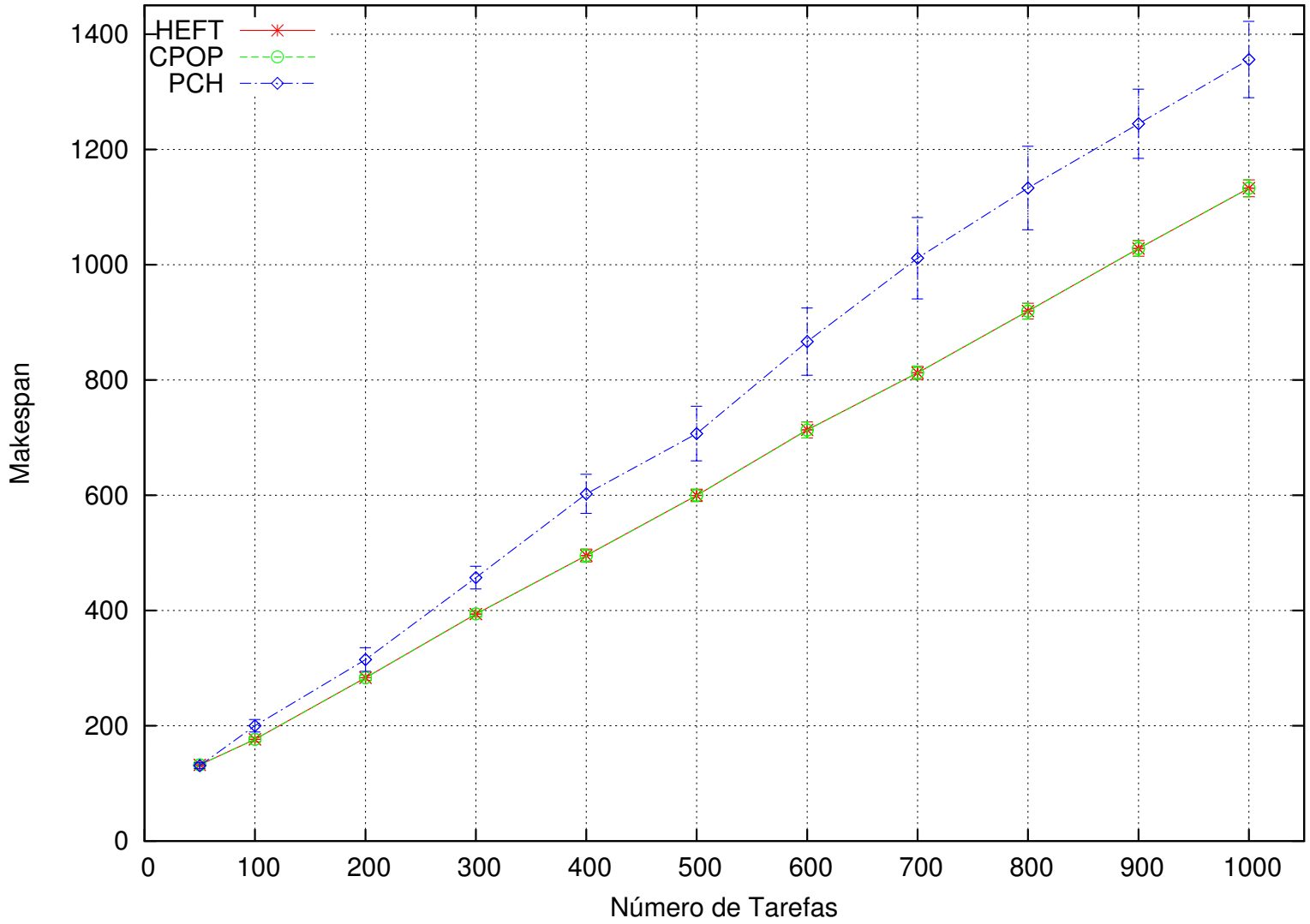
Desempenho - GridPP



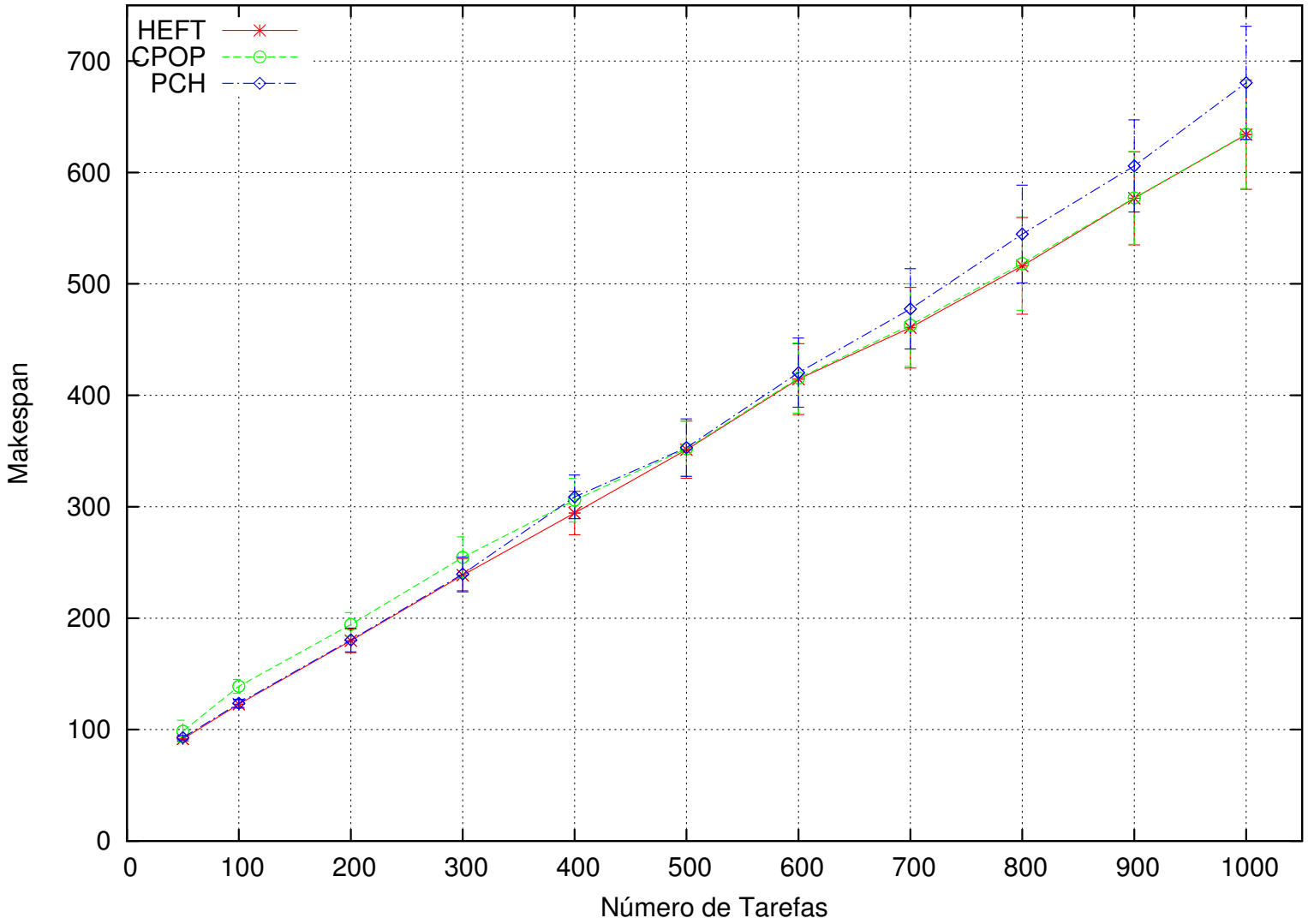
Escalabilidade: (i) Montage - SmallGrid Homogênea



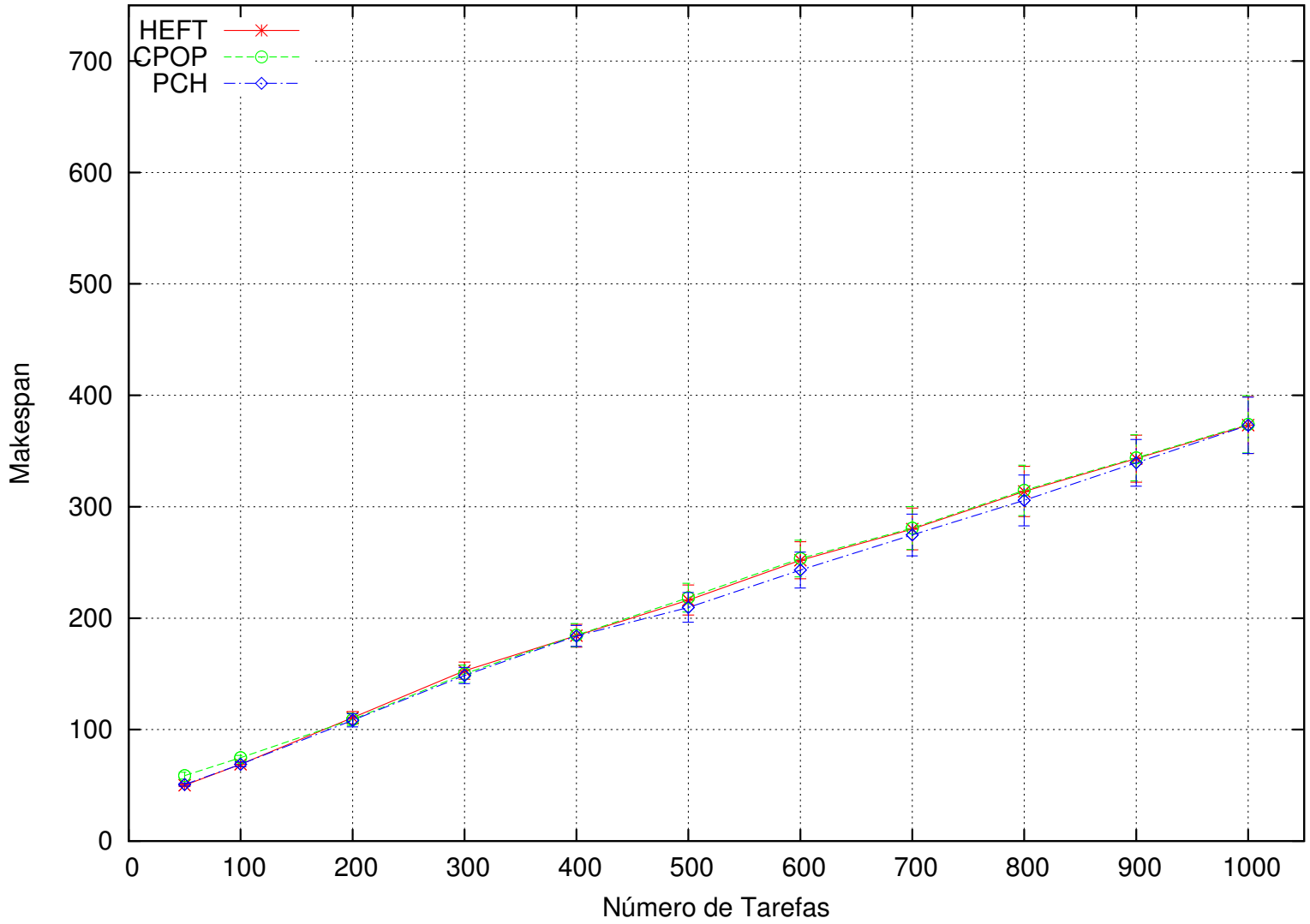
Escalabilidade: (i) Montage - SmallGrid Heterogênea



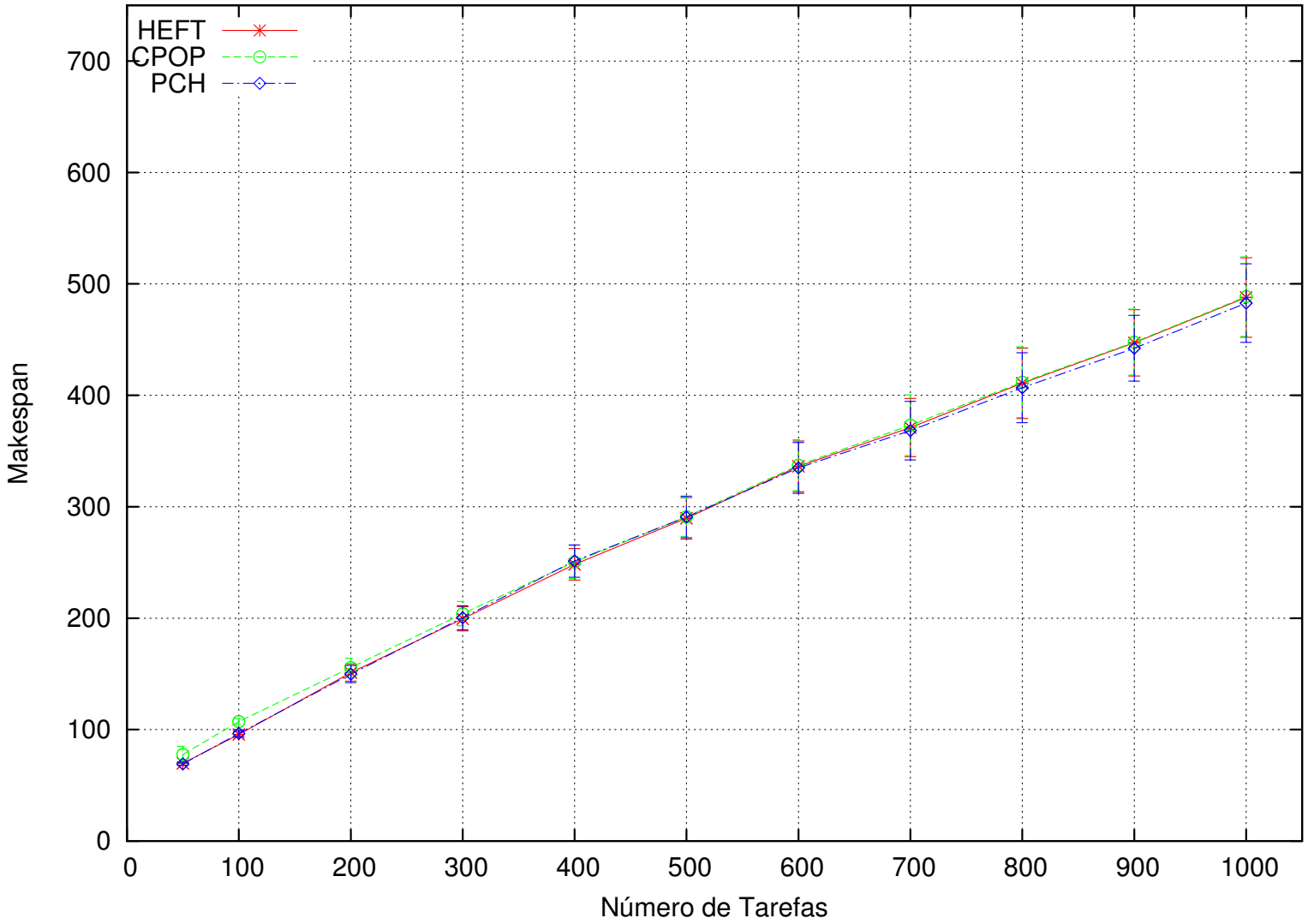
Escalabilidade: (i) Montage - DAS-3



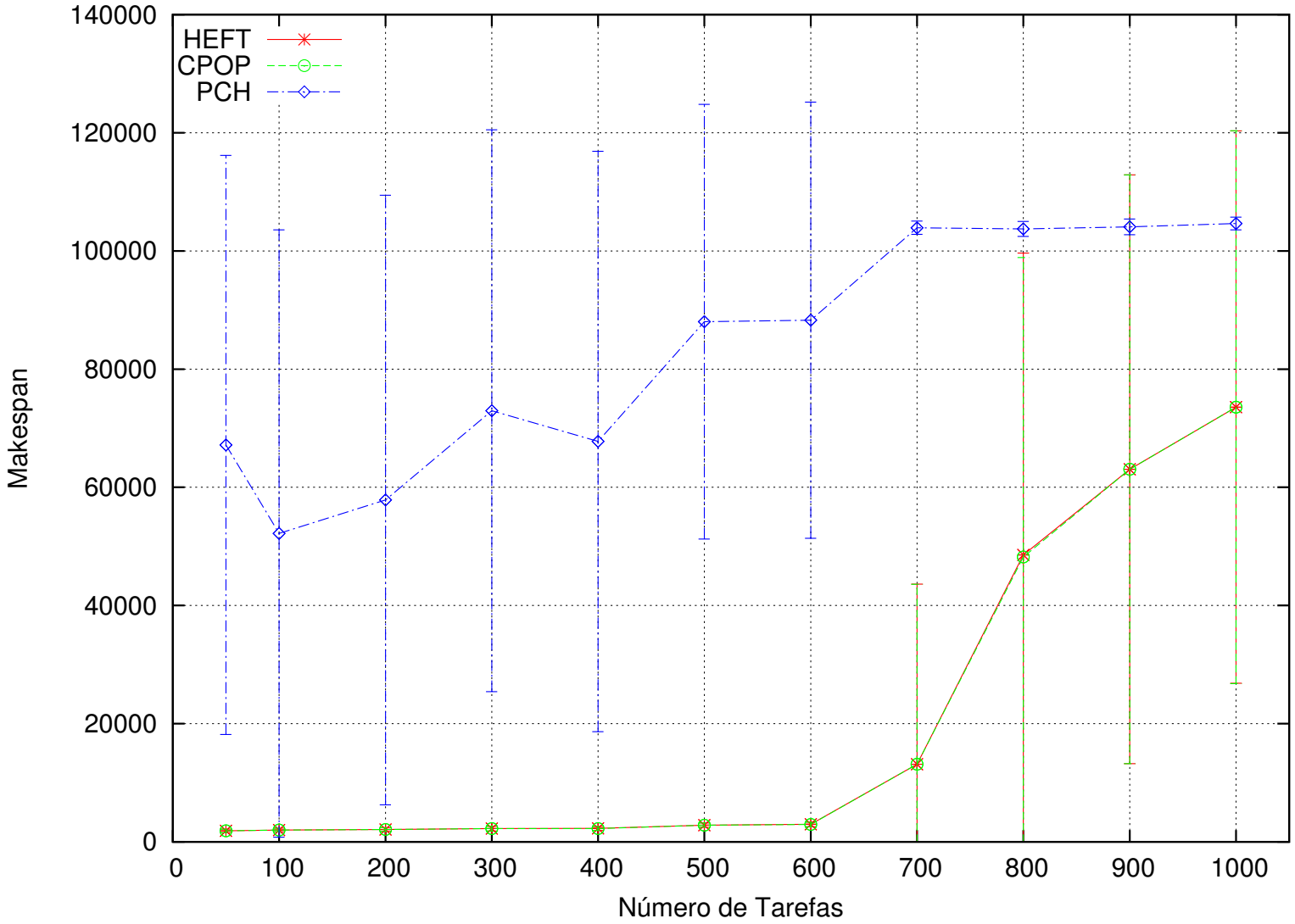
Escalabilidade: (i) Montage - Grid5000



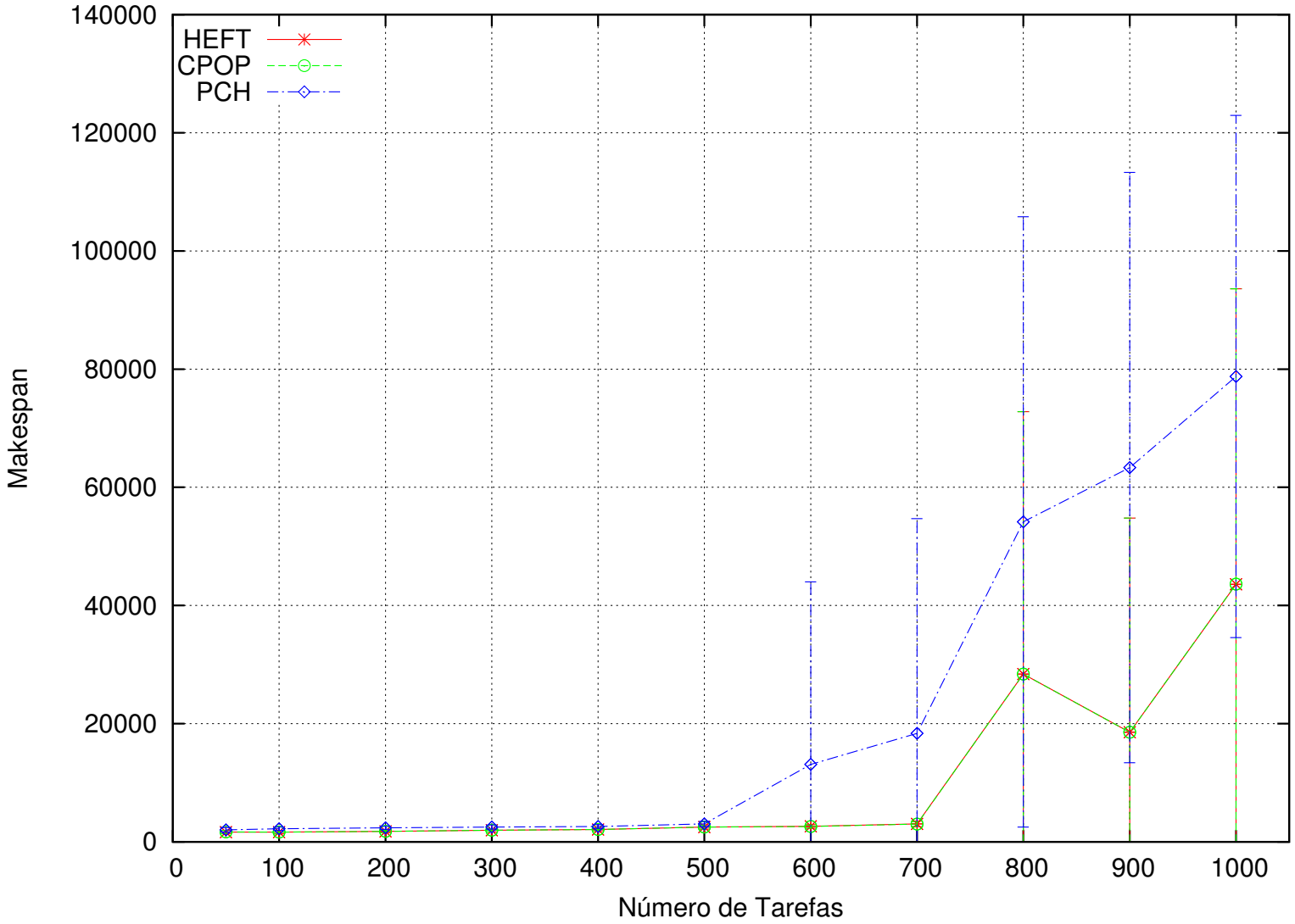
Escalabilidade: (i) Montage - GridPP



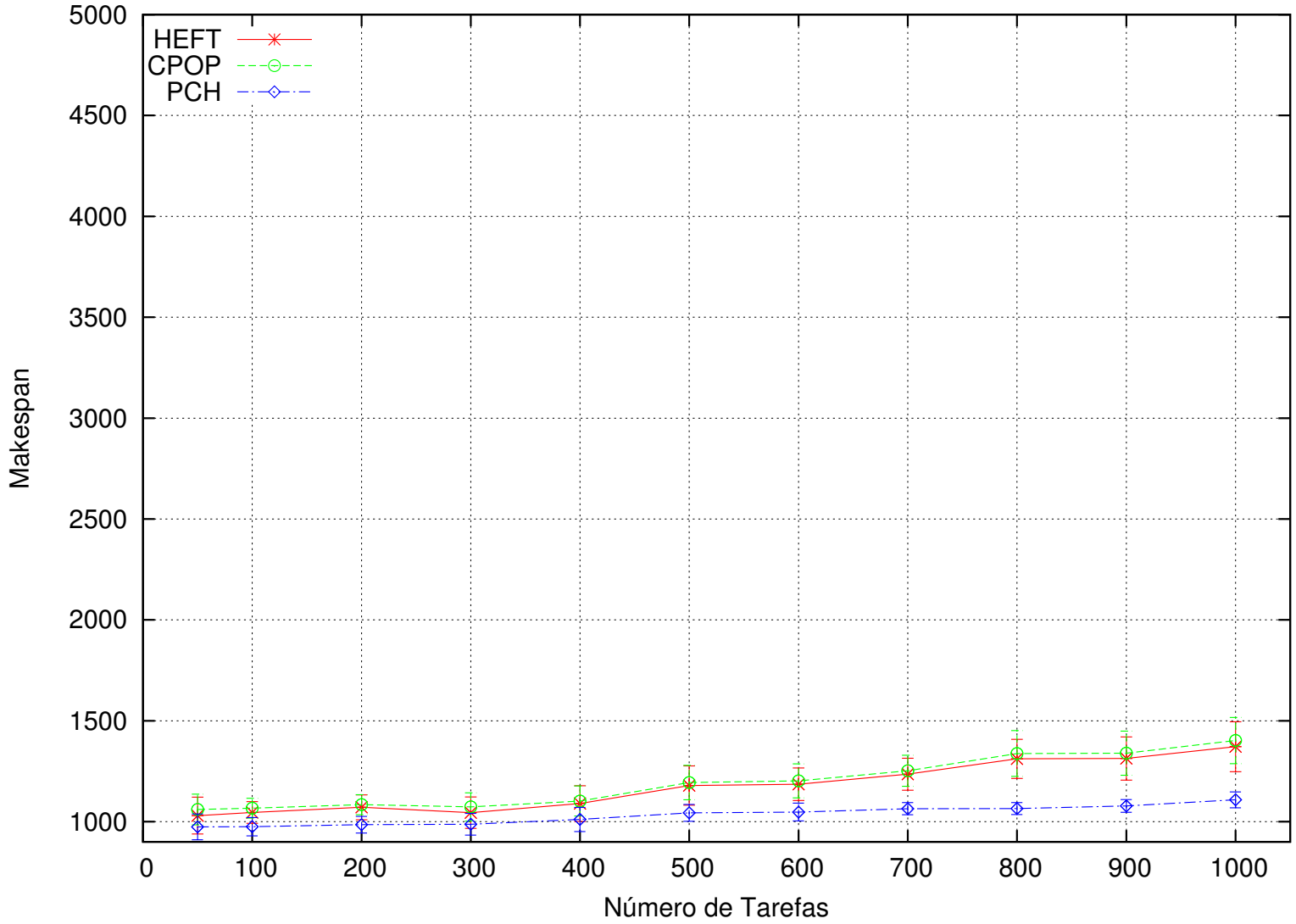
Escalabilidade: (i) CyberShake - SmallGrid Homogênea



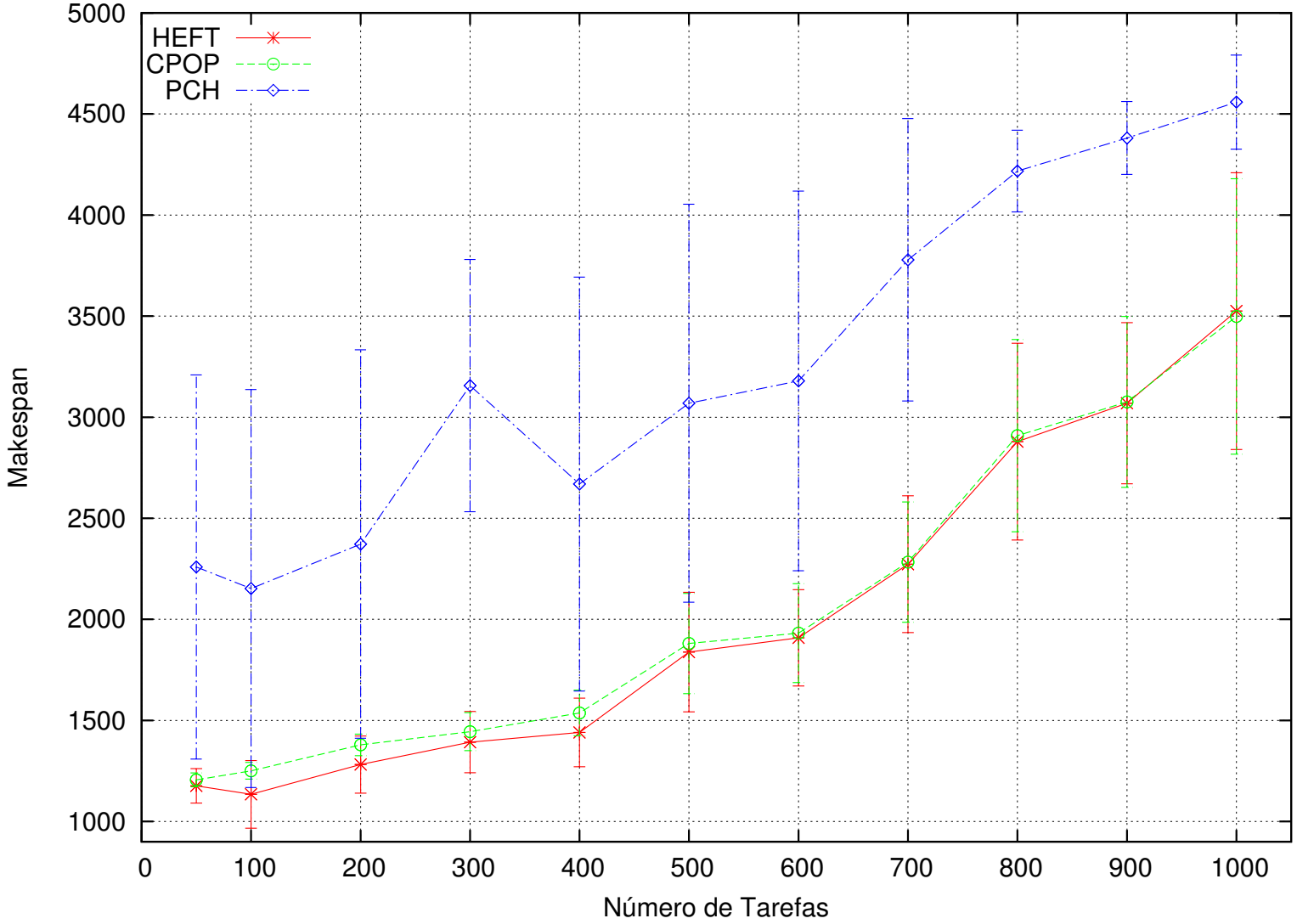
Escalabilidade: (i) CyberShake - SmallGrid Heterogênea



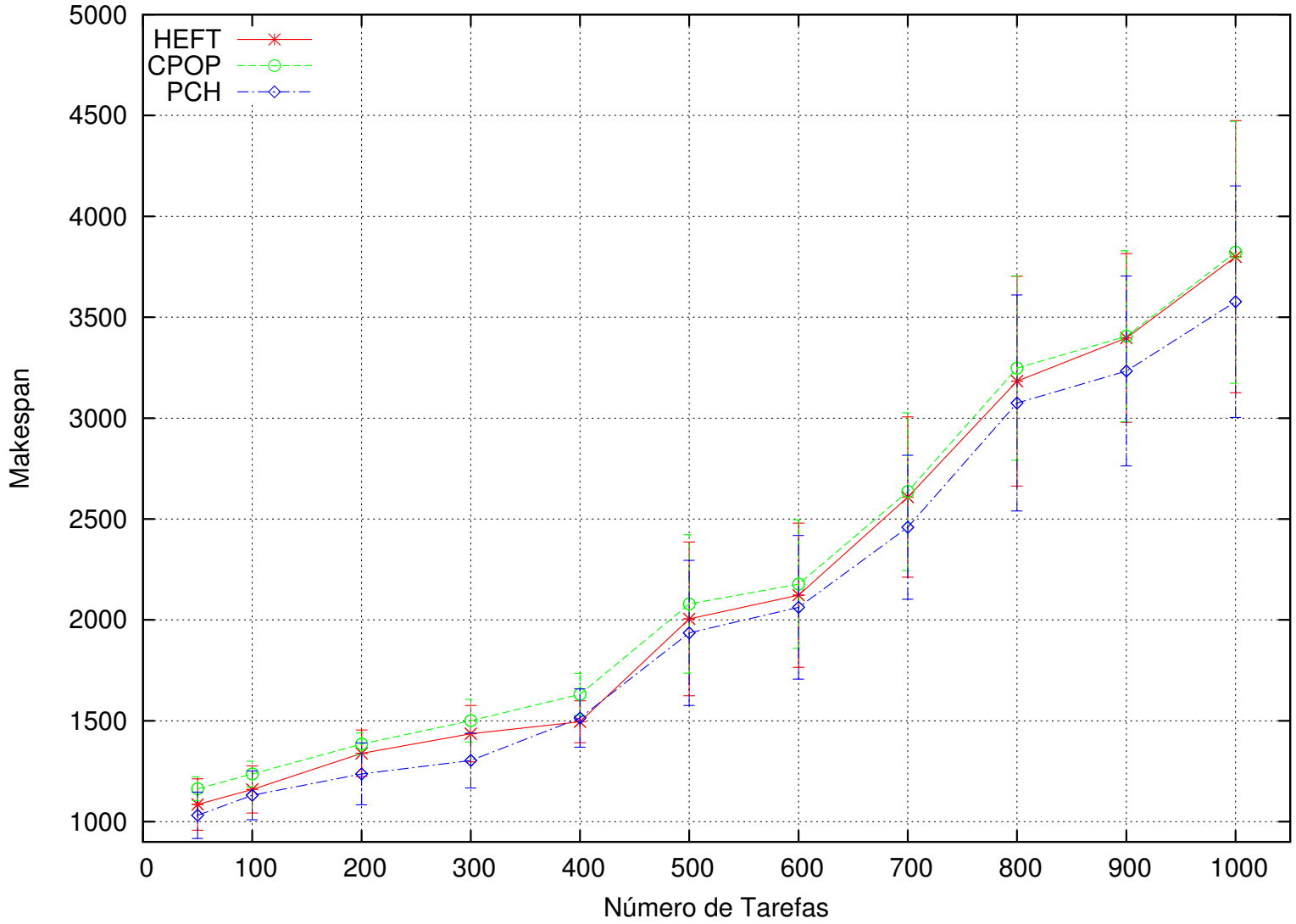
Escalabilidade: (i) CyberShake - DAS-3



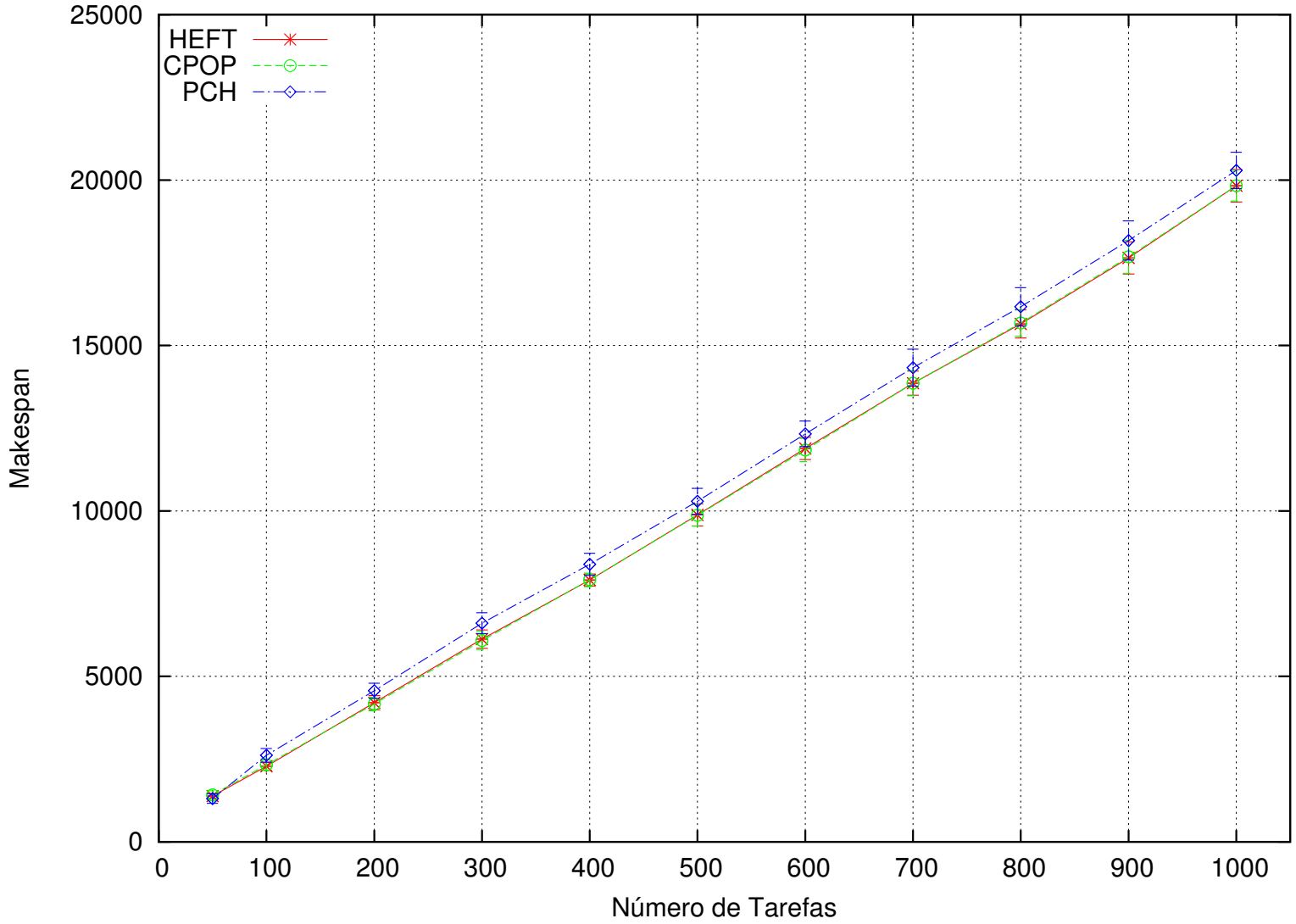
Escalabilidade: (i) CyberShake - Grid5000



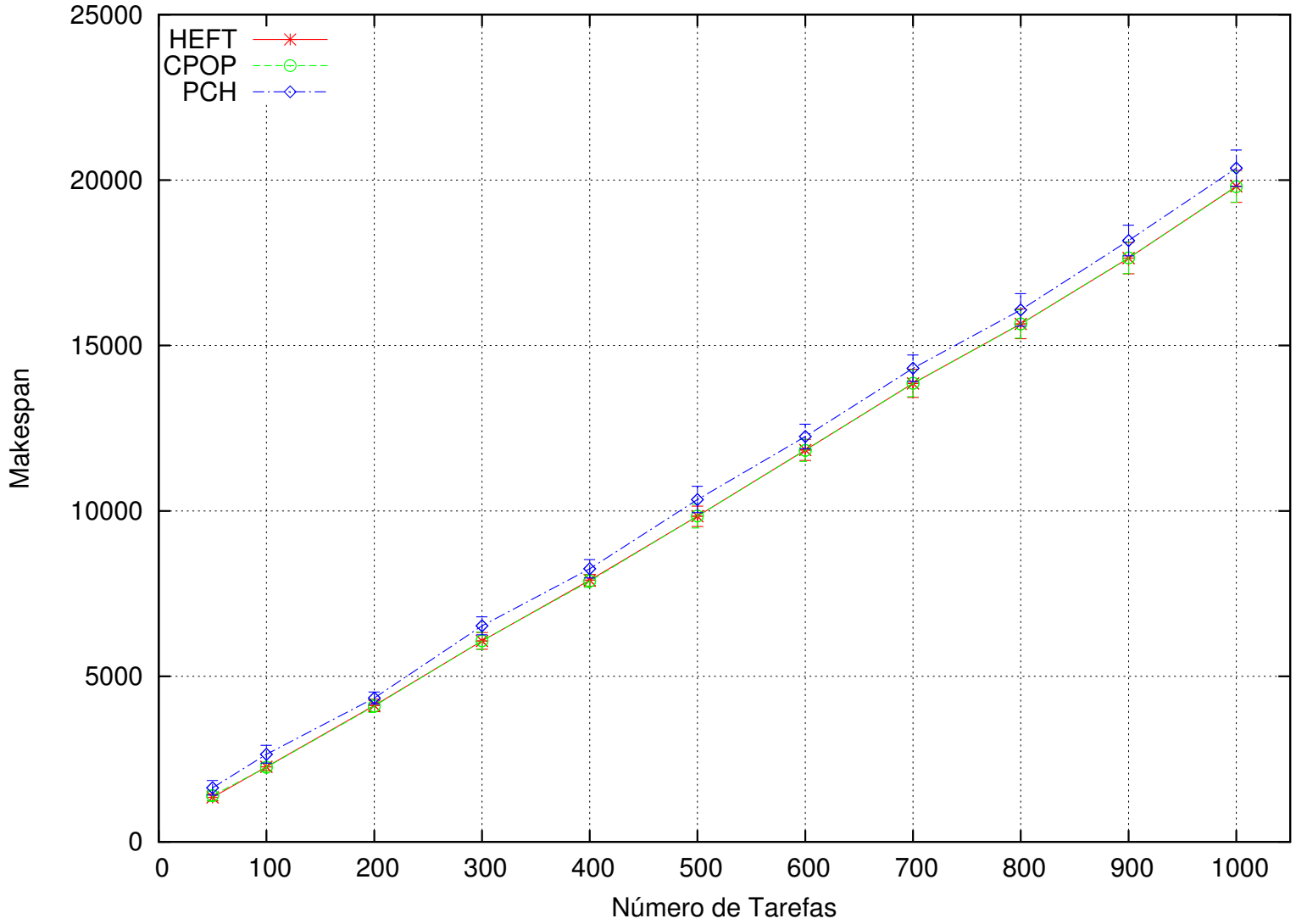
Escalabilidade: (i) CyberShake - GridPP



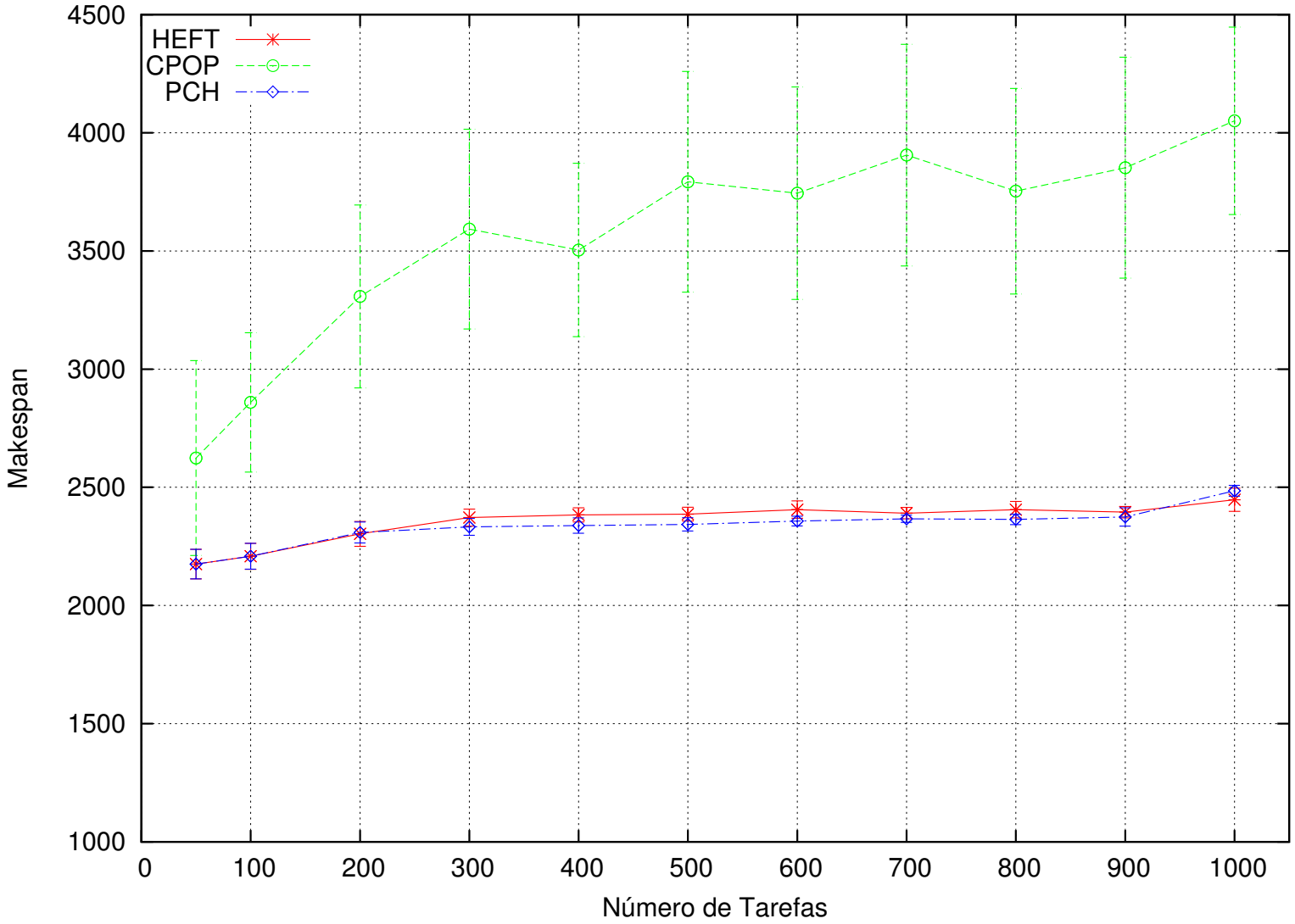
Escalabilidade: (i) Ligo - SmallGrid Homogênea



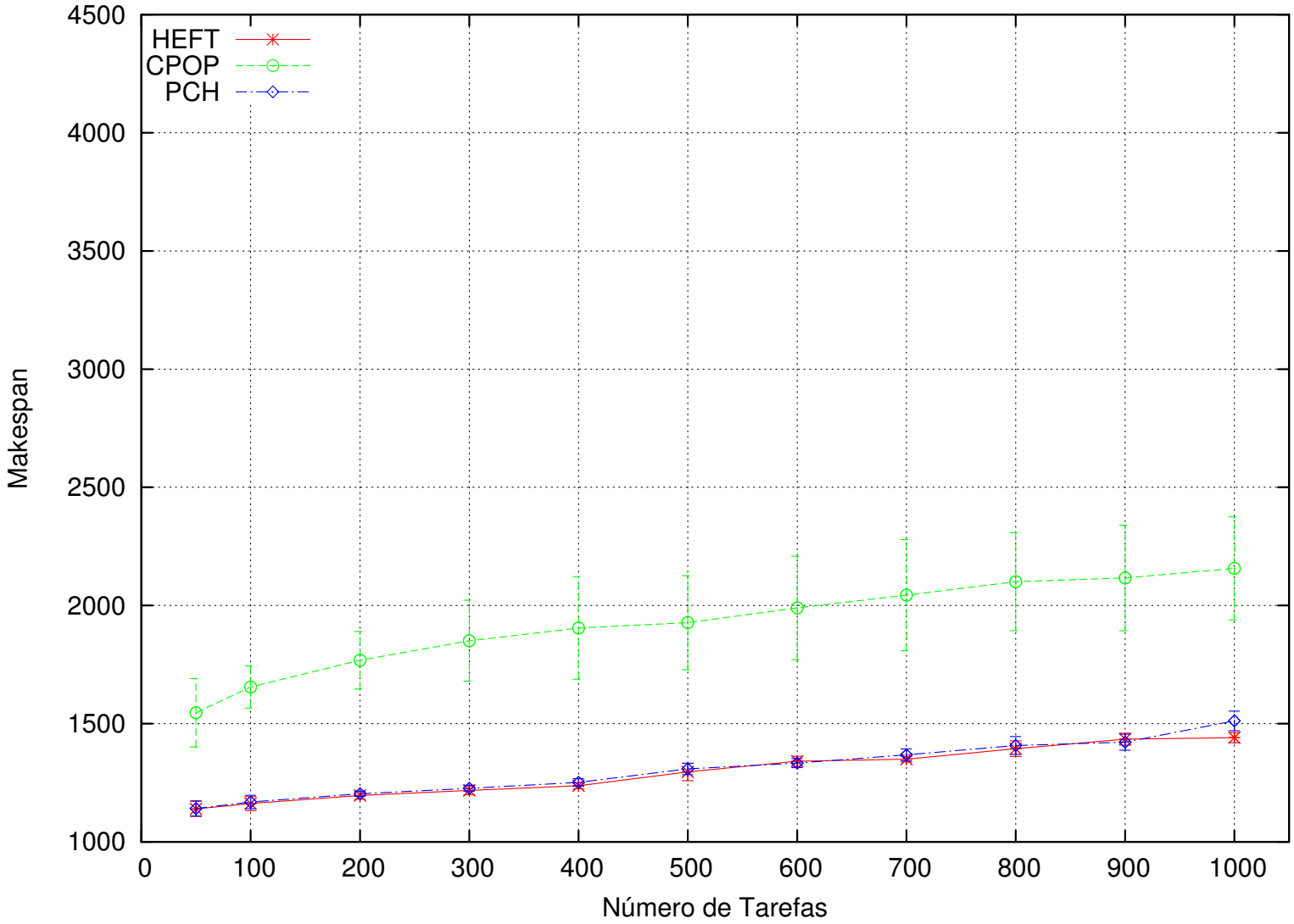
Escalabilidade: (i) Ligo - SmallGrid Heterogênea



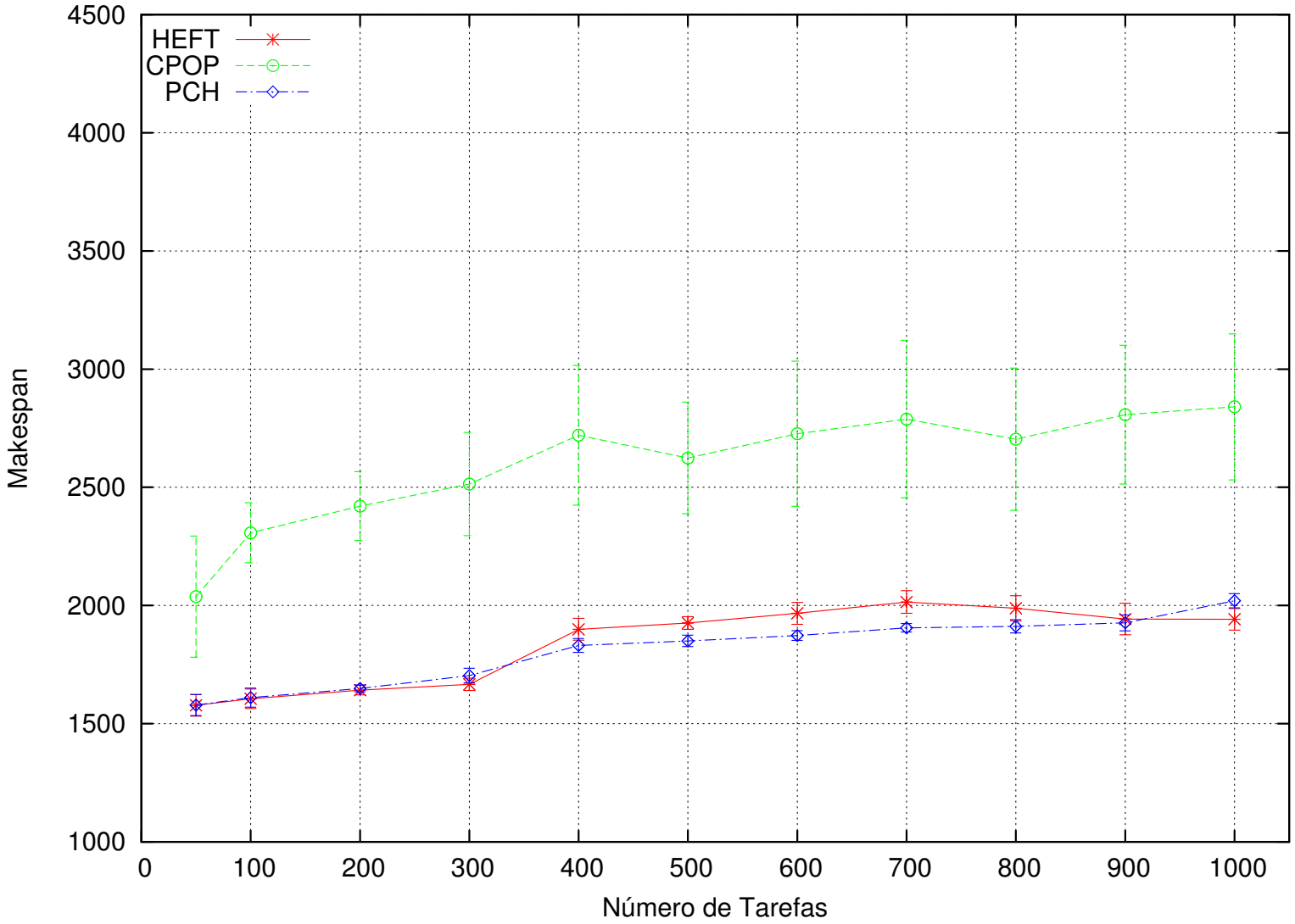
Escalabilidade: (i) Ligo - DAS-3



Escalabilidade: (i) Ligo - Grid5000



Escalabilidade: (i) Ligo - GridPP



Adaptabilidade

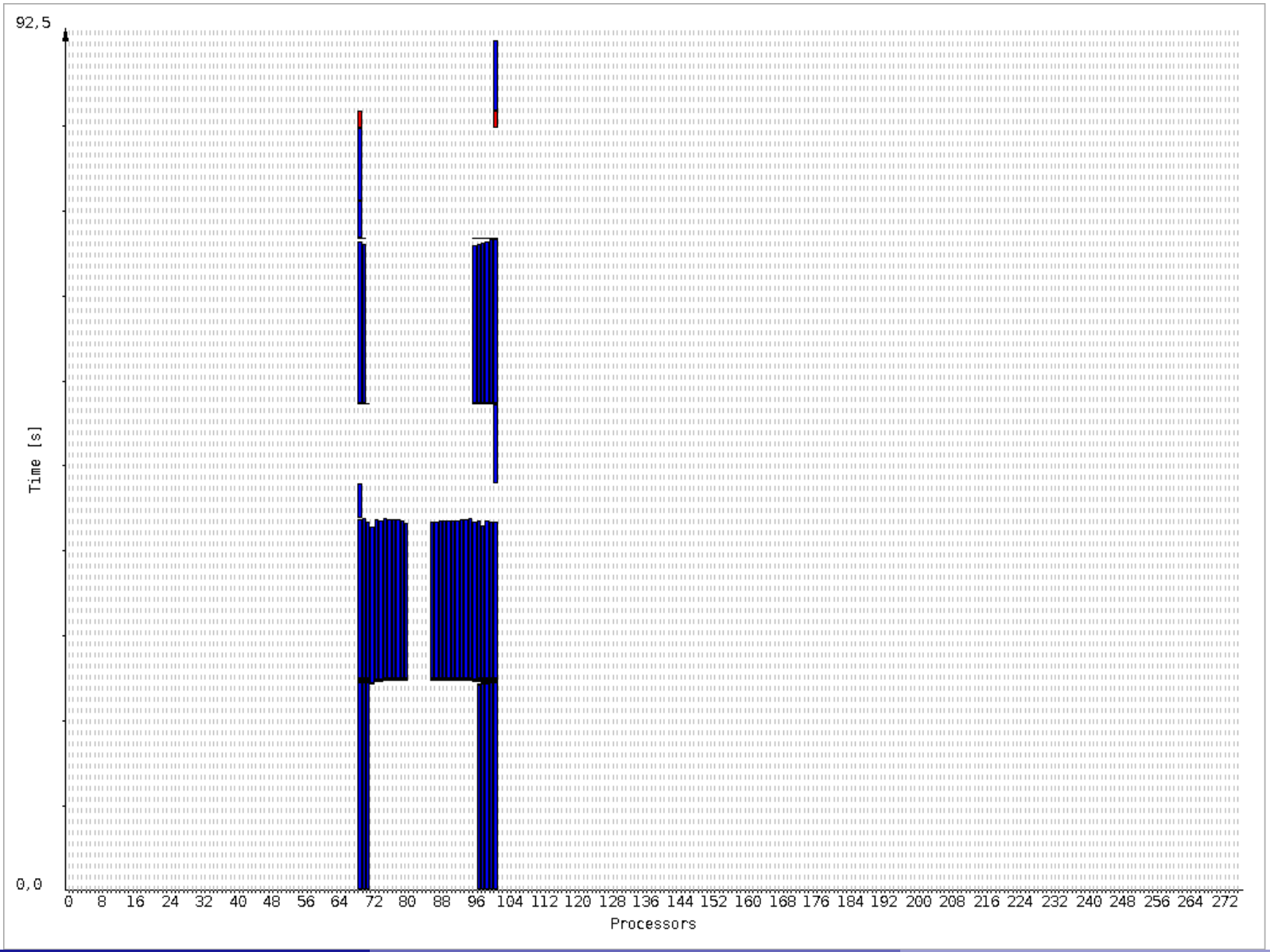
HEFT	(Das3, G5k)	(G5k, Gpp)
Montage	1,65	0,76
CyberShake	0,59	0,93
Epigenomics	1,68	0,74
Ligo	1,82	0,70

CPOP	(Das3, G5k)	(G5k, Gpp)
Montage	1,67	0,75
CyberShake	0,59	0,92
Epigenomics	1,66	0,76
Ligo	1,85	0,74

PCH	(Das3, G5k)	(G5k, Gpp)
Montage	1,75	0,75
CyberShake	0,32	1,59
Epigenomics	1,67	0,77
Ligo	1,79	0,72

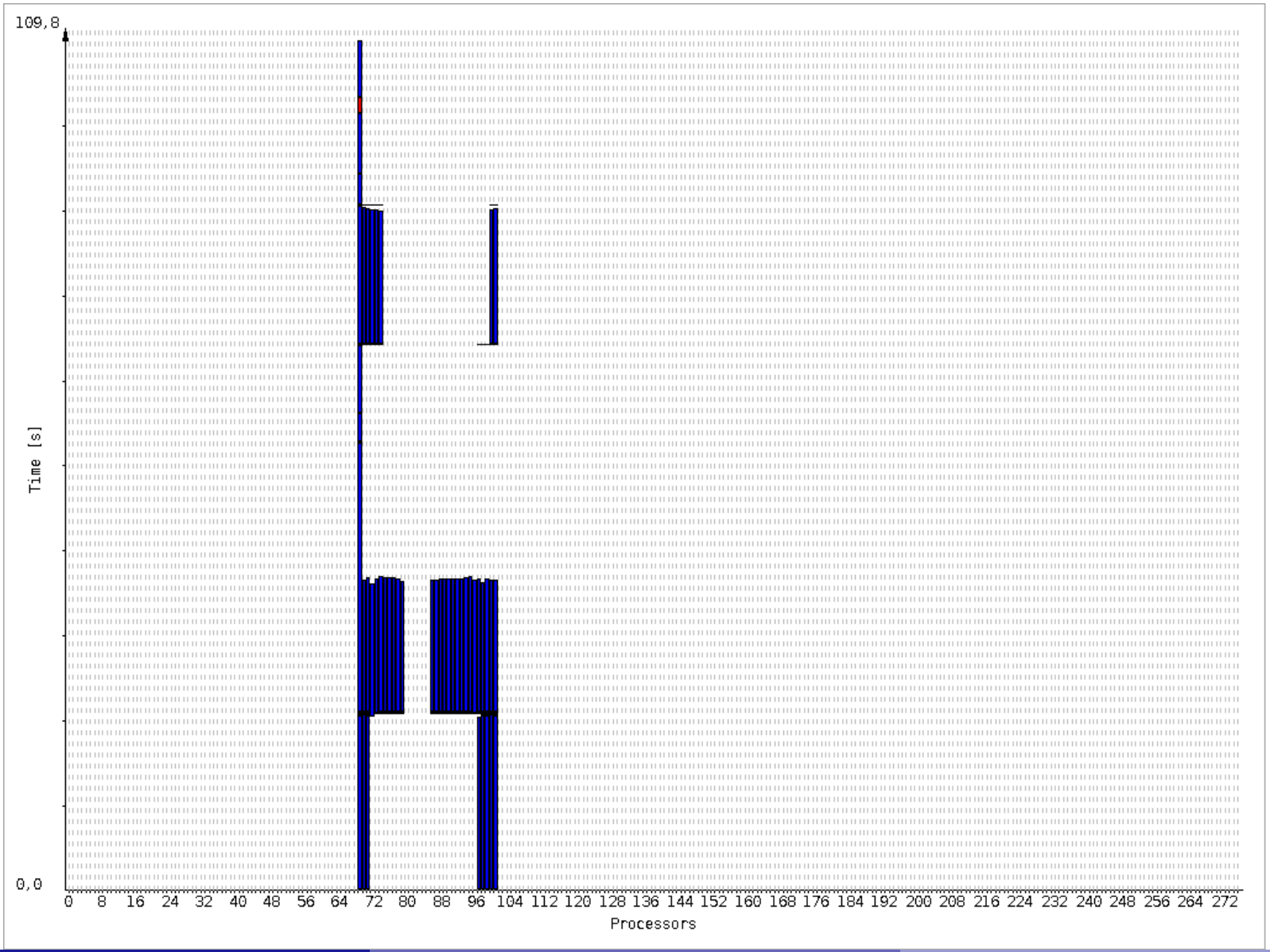
Distribuição da Carga de Trabalho

Montage - DAS-3 - HEFT



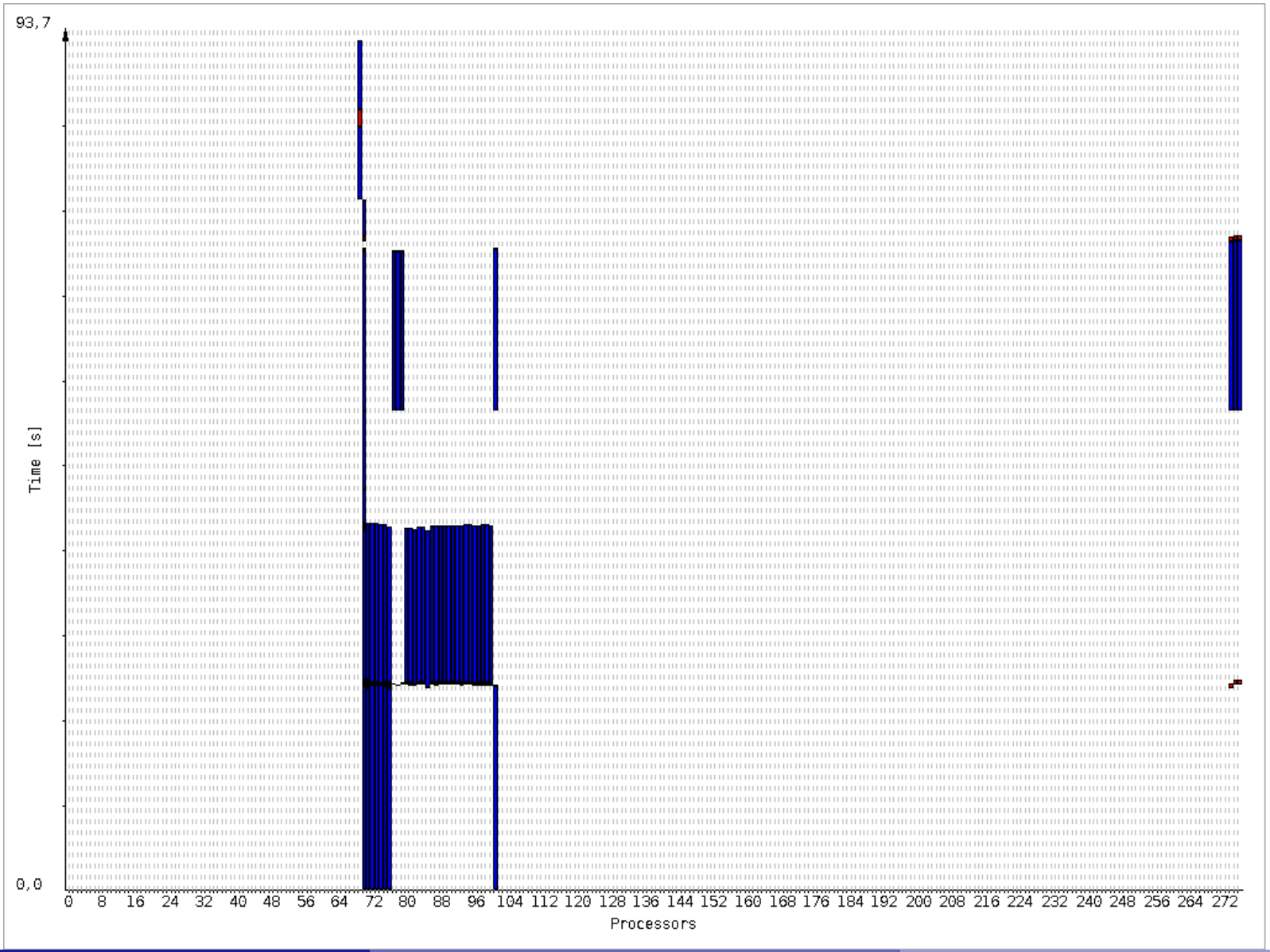
Distribuição da Carga de Trabalho

Montage - DAS-3 - CPOP



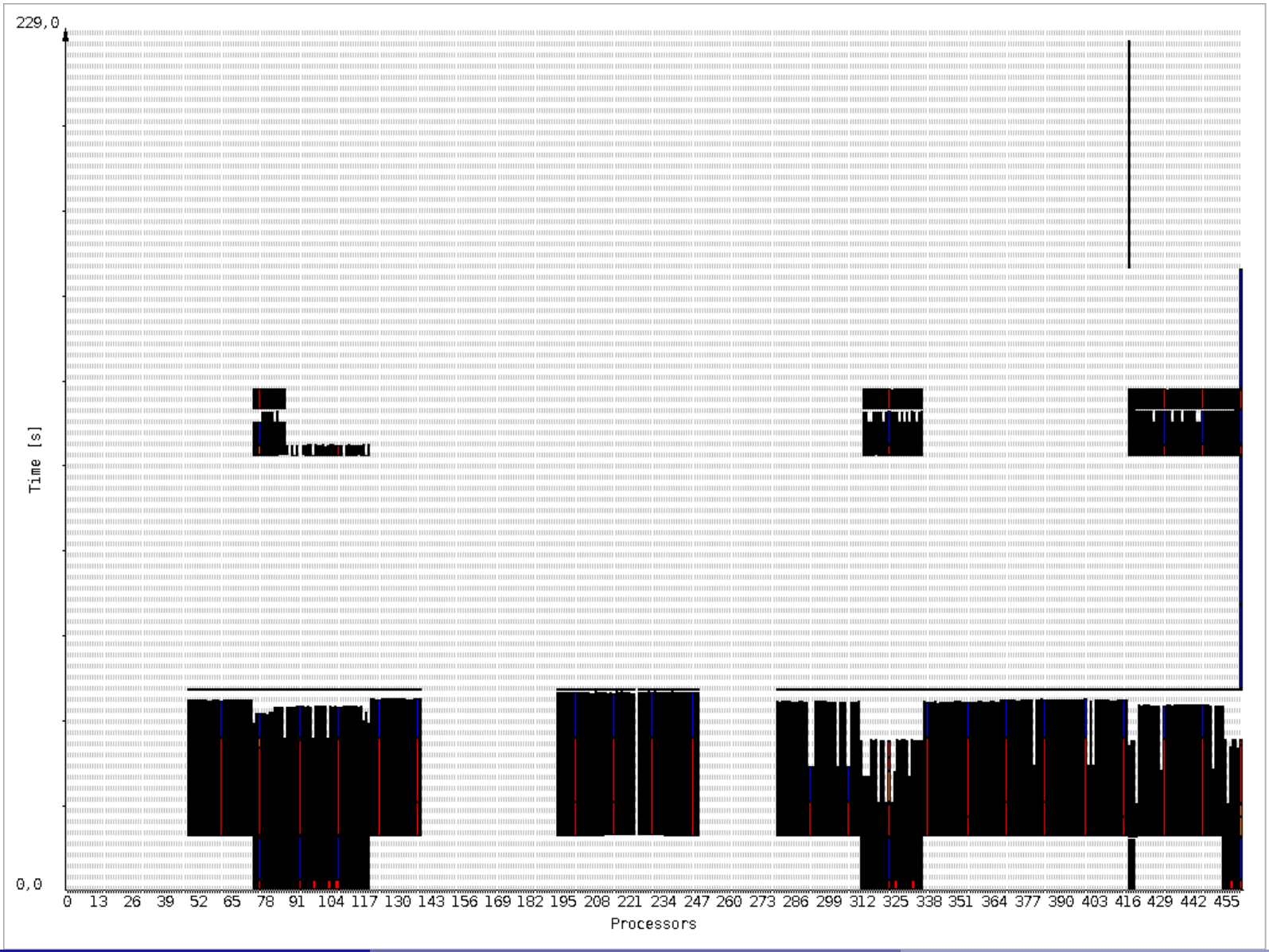
Distribuição da Carga de Trabalho

Montage - DAS-3 - PCH



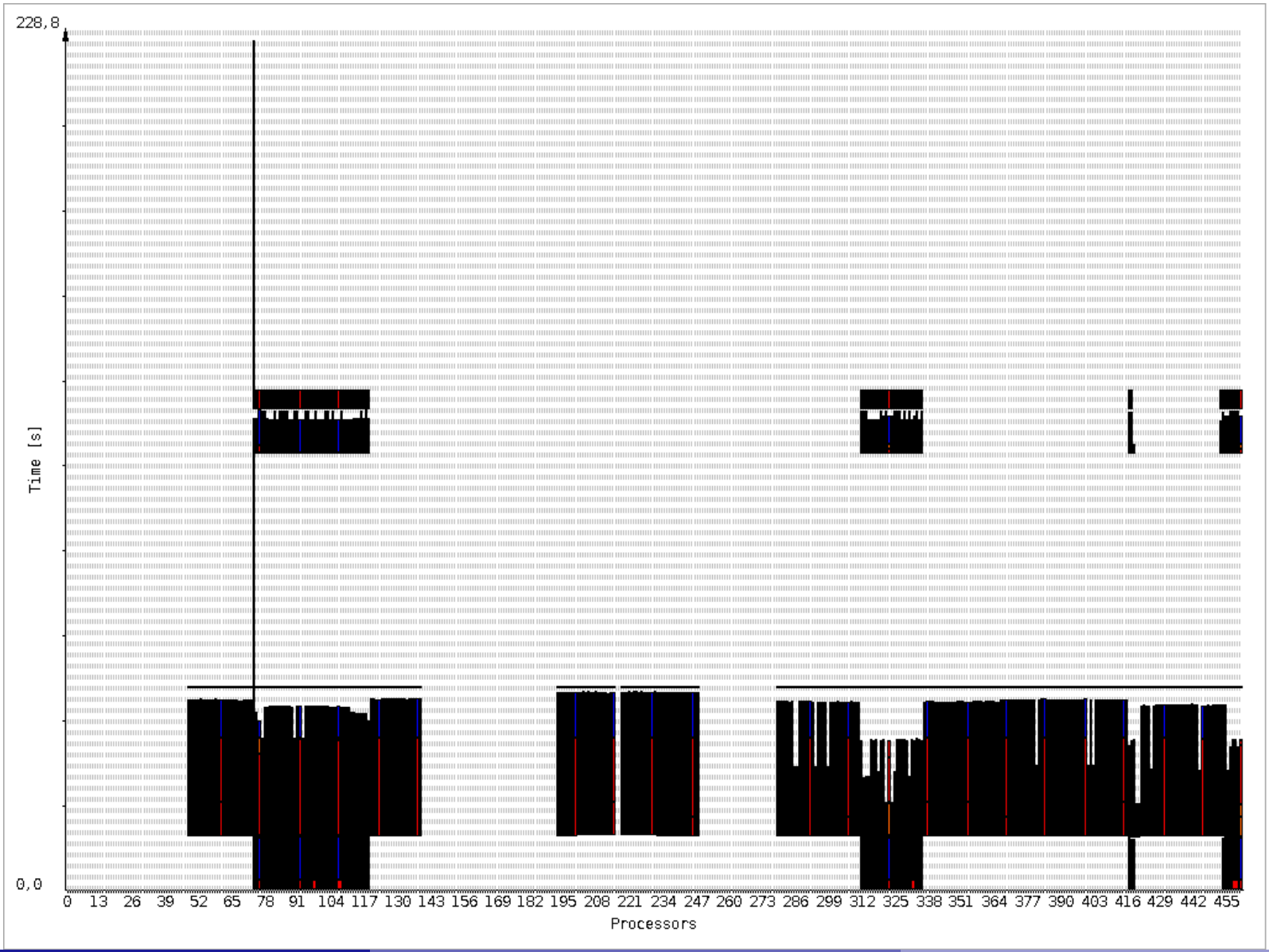
Distribuição da Carga de Trabalho

Montage - Grid5000 - HEFT



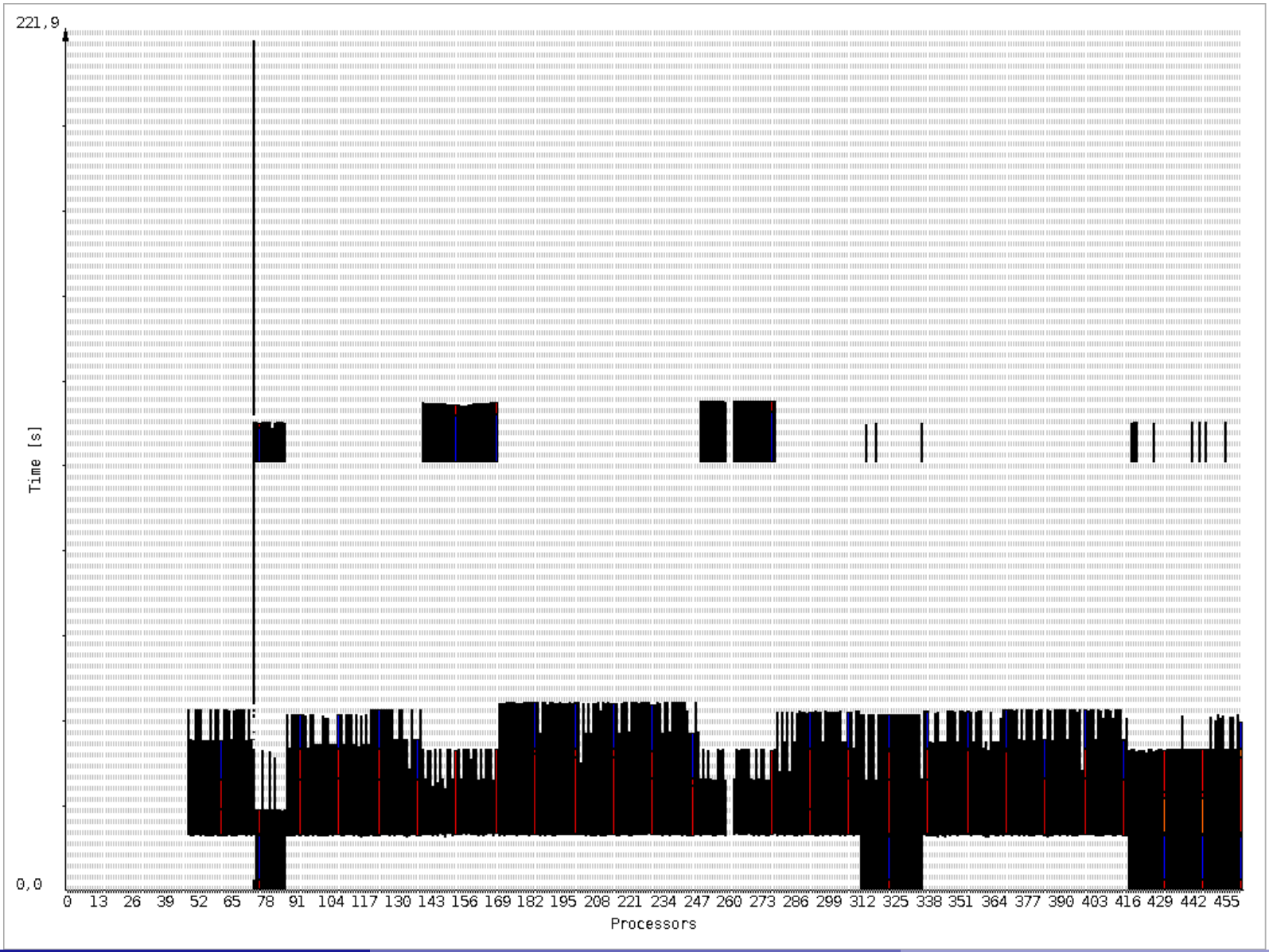
Distribuição da Carga de Trabalho

Montage - Grid5000 - CPOP



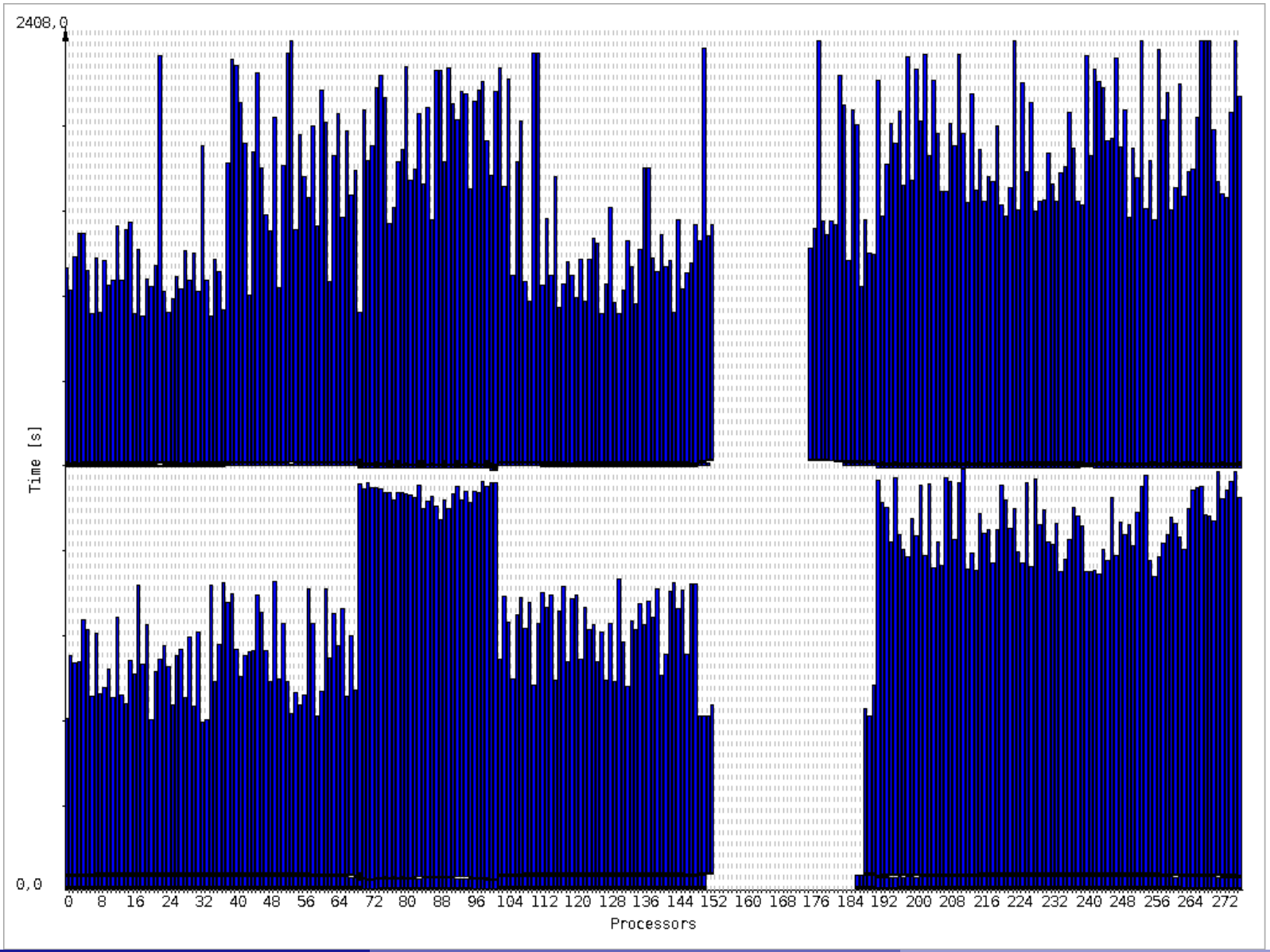
Distribuição da Carga de Trabalho

Montage - Grid5000 - PCH



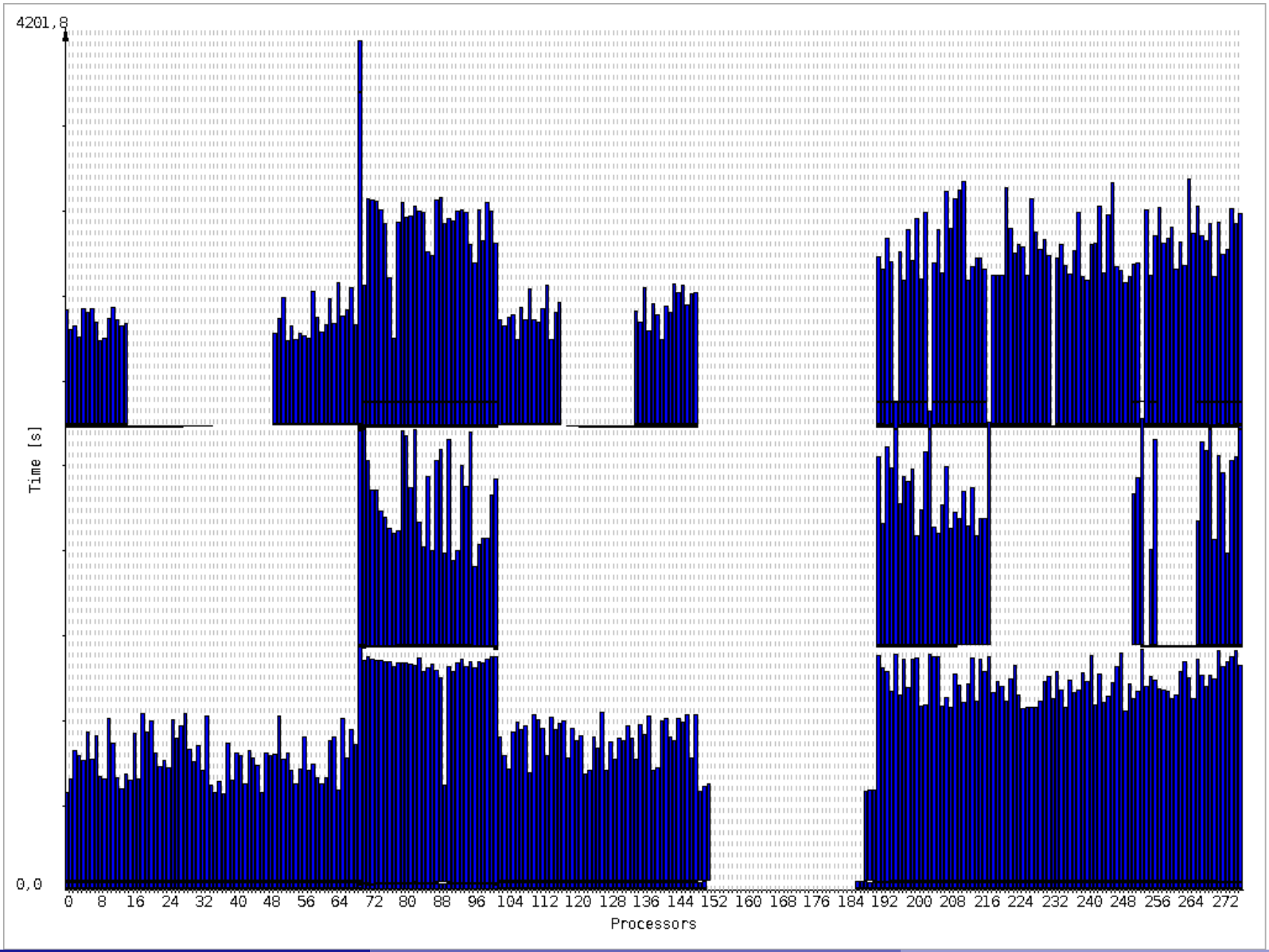
Distribuição da Carga de Trabalho

Ligo - DAS-3 - HEFT



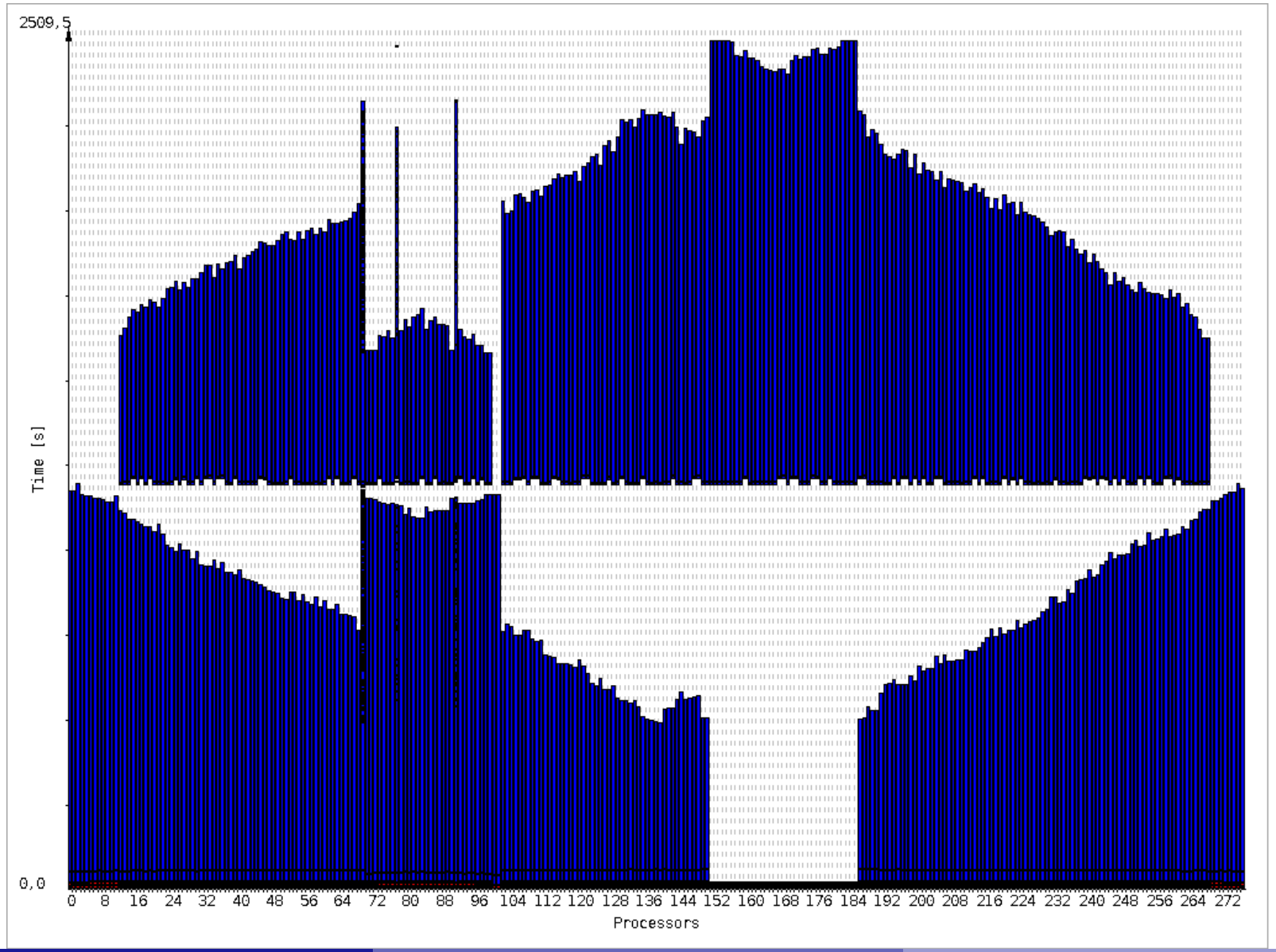
Distribuição da Carga de Trabalho

Ligo - DAS-3 - CPOP



Distribuição da Carga de Trabalho

Ligo - DAS-3 - PCH



Conclusões

- Na literatura foram propostos diferentes algoritmos de escalonamento
- A escolha de um algoritmo de escalonamento que tenha as características necessárias para obter um desempenho bom em um determinado cenário é indispensável
- Ao comparar algoritmos de escalonamento deve seguir principalmente quatro critérios:
  - Desempenho
  - Escalabilidade
  - Adaptabilidade
  - Distribuição da Carga do Trabalho
- É importante entender e saber qual é o tipo de aplicação, pode ser de dois tipos: Aplicação Regular e Aplicação Irregular
- Em aplicações irregulares é mais difícil medir escalabilidade

Conclusões

- O algoritmo HEFT possui um bom desempenho na maioria dos casos, apresentando uma estabilidade
- Os algoritmos CPOP e PCH, apresentaram um desempenho bom sobre determinadas circunstâncias
- No caso do algoritmo CPOP possui uma dependência sobre a estrutura da aplicação e da arquitetura, dado que escalona as tarefas do caminho crítico
- O algoritmo PCH agrupa as tarefas e escalona cada grupo no processador que oferece o melhor tempo de término. Esse critério perde sentido em tipos de aplicações com tarefas de sincronização crítica

Contribuições

1. Classificação dos tipos de aplicações para grade com tarefas dependentes: (i) regulares e (ii) irregulares
2. Uma metodologia para fazer comparação de algoritmos de escalonamento, baseado em determinadas configurações e métricas
3. Atualização, modelagem e especificação para a simulação das arquiteturas para grade: (i) DAS-3, (ii) Grid5000 e (iii) GridPP, sobre o simulador SimGrid v3.5
4. Repositório de imagens dos resultados do escalonamento, criadas nas simulações dos algoritmos

Trabalhos Futuros

- Sugerimos ter um maior conjunto de aplicações, tanto em tamanho quanto em forma da estrutura. Como uma alternativa existe o uso de um gerador randômico de grafos de aplicações
- Arquiteturas com processadores com vários núcleos, este tipo de experimentos não foi abordado pelo fato do simulador ainda não suportar este tipo de arquiteturas

Muito Obrigado

- Orientador Alfredo Goldman
- Banca: Profa. Dra. Liria Matsumoto Sato e Prof. Dr. Philippe Navaux
- Projeto SimGrid e Projeto Pegasus
- Família e amigos ( ,  ,  , )
- Colegas do LCPD
- Ao IME(professores e funcionários).