SSC-0158
Computação em Nuvem

Aula 03 – Escopo

Prof. Julio Cezar Estrella
jcezar@icmc.usp.br

Créditos

Os slides integrantes deste material foram construídos a partir dos conteúdos relacionados às referências bibliográficas descritas neste documento.

Todos já devem ter ouvido algo sobre Cloud Computing

- Ou ao menos algumas dessas ideias:
  - “Computação em nuvem finalmente tornou realidade o sonho da computação utilitária”
  - “Desenvolvedores não precisam mais se preocupar em conseguir grandes somas de dinheiro antes de colocar uma nova aplicação web no ar”
  - “Adeus aos problemas de provisionamento de servidores (Elasticidade dos recursos)”
  - Software como um serviço Plataforma como um serviço
  - Infraestrutura como um serviço

Histórico e motivações

Problemas que (ainda) requerem constante inovação tecnológica

- Problemas “em escala da web”
- **Grandes** *data centers*
- Computação paralela e
- distribuída Aplicações web interativas

Problemas em escala da web

- Características
  - Definitivamente *data-intensive*
  - Mas podem também ser *processing-intensive*
- Exemplos:
  - *Crawling*, indexação, busca, mineração de dados da web
  - Pesquisa em biologia computacional na era “pós-genômica”
  - Processamento de dados científicos (física, astronomia,
  - etc.) Redes de sensores
  - Aplicações Web 2.0 etc.

De qual volume de dados estamos falando?

Problemas da ordem de petabytes!

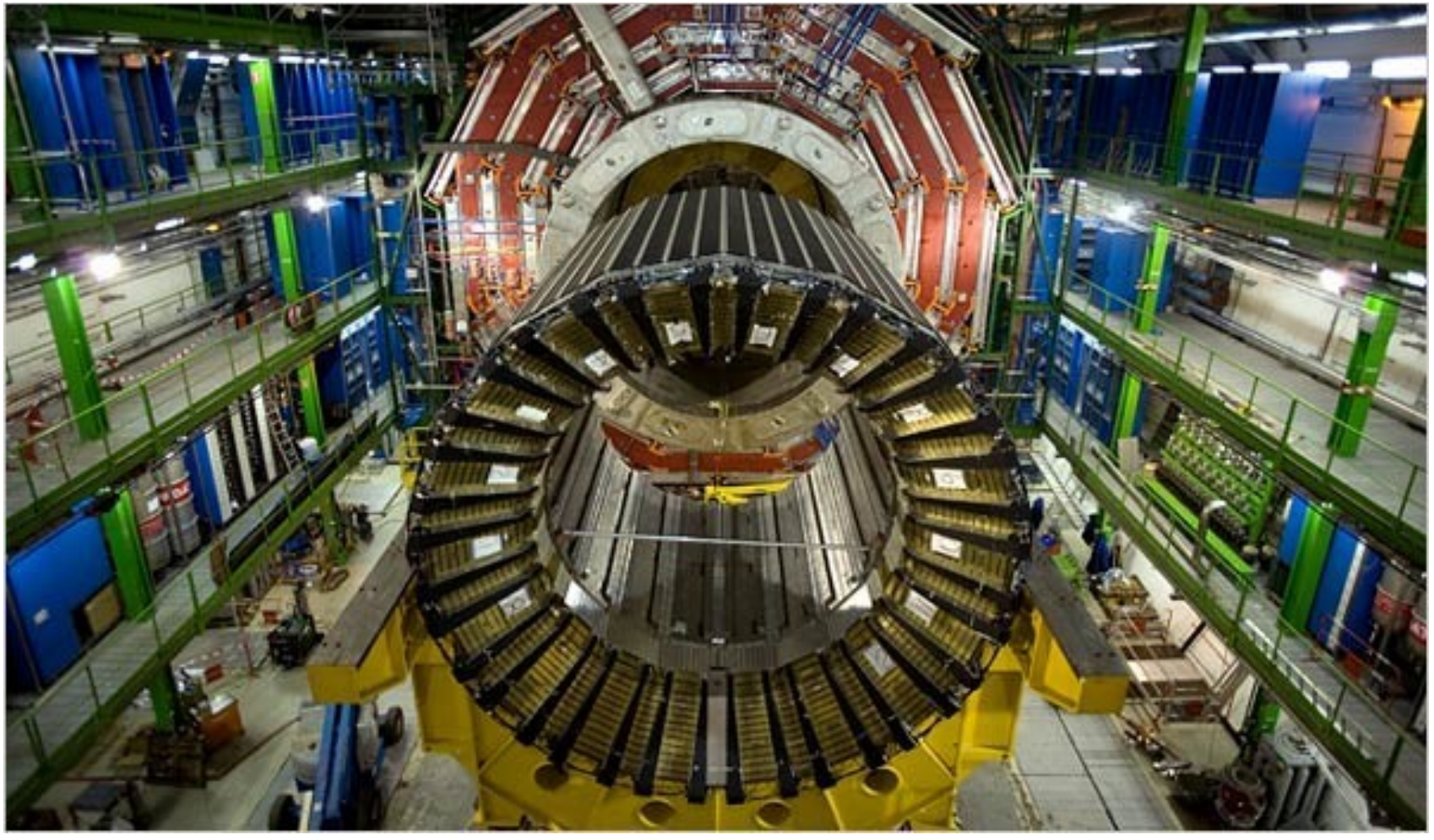
1 PB =
1.000.000.000.000.000 B
$= 1.000^5$ B
$= 10^{15}$ B
= 1 milhão de gigabytes
= 1 mil terabytes

De qual volume de dados estamos falando?

Muitos, mas **muitos** dados

- O Google processa cerca de **20 petabytes** de dados por dia (2008)
- O Wayback Machine tem cerca de **3 petabytes + 100 terabytes/dia** (mar/2009)
- O Facebook tem cerca de **2,5 petabytes de dados de usuários + 15 terabytes/dia** (abr/2009)
- O site eBay tem cerca de **6,5 petabytes de dados dos usuários + 50 terabytes/dia** (mai/2009)
- O Grande Colisor de Hádrons do CERN irá gerar cerca de **15 petabytes/ano**

De qual volume de dados estamos falando?



De qual volume de dados estamos falando?

Problemas da ordem de petabytes!

$1\ PB = 1.000.000.000.000.000\ B$

$= 1.000^5\ B$

$= 10^{15}\ B$

$=$ 1 milhão de gigabytes

$=$ 1 mil terabytes

De qual volume de dados estamos falando?

$$1\ \text{PB} = 1.000.000.000.000.000\ \text{B}$$
$$= 1.000^5\ \text{B}$$
$$= 10^{15}\ \text{B}$$
$$= 1\ \text{milhão de gigabytes}$$
$$= 1\ \text{mil terabytes}$$

Ou seja, os 15 petabytesque o CERN irá gerar por ano equivalem a 15 milhões de gigabytes. Seriam necessários1,7 milhão de DVDs *dual-layer* para armazenar tanta informação!

O que se faz com tantos dados?

- Encontram informações sobre novos fatos Casamento de padrões com informações da web ex: quem matou John Lennon?
- Procuram por novas relações entre os dados
  - Alguns padrões levam a novas relações:
    - os fatos: “Nascimento-de(Mozart, 1756)” e “Nascimento-de(Einstein, 1879)”
    - levam aos dados: “Wolfgang Amadeus Mozart (1756–1791)” e “Einstein nasceu em 1879”
    - que levam a diferentes padrões: “PESSOA (DATA –” e “PESSOA nasceu em DATA”
    - que, por sua vez, permitem encontrar novos fatos

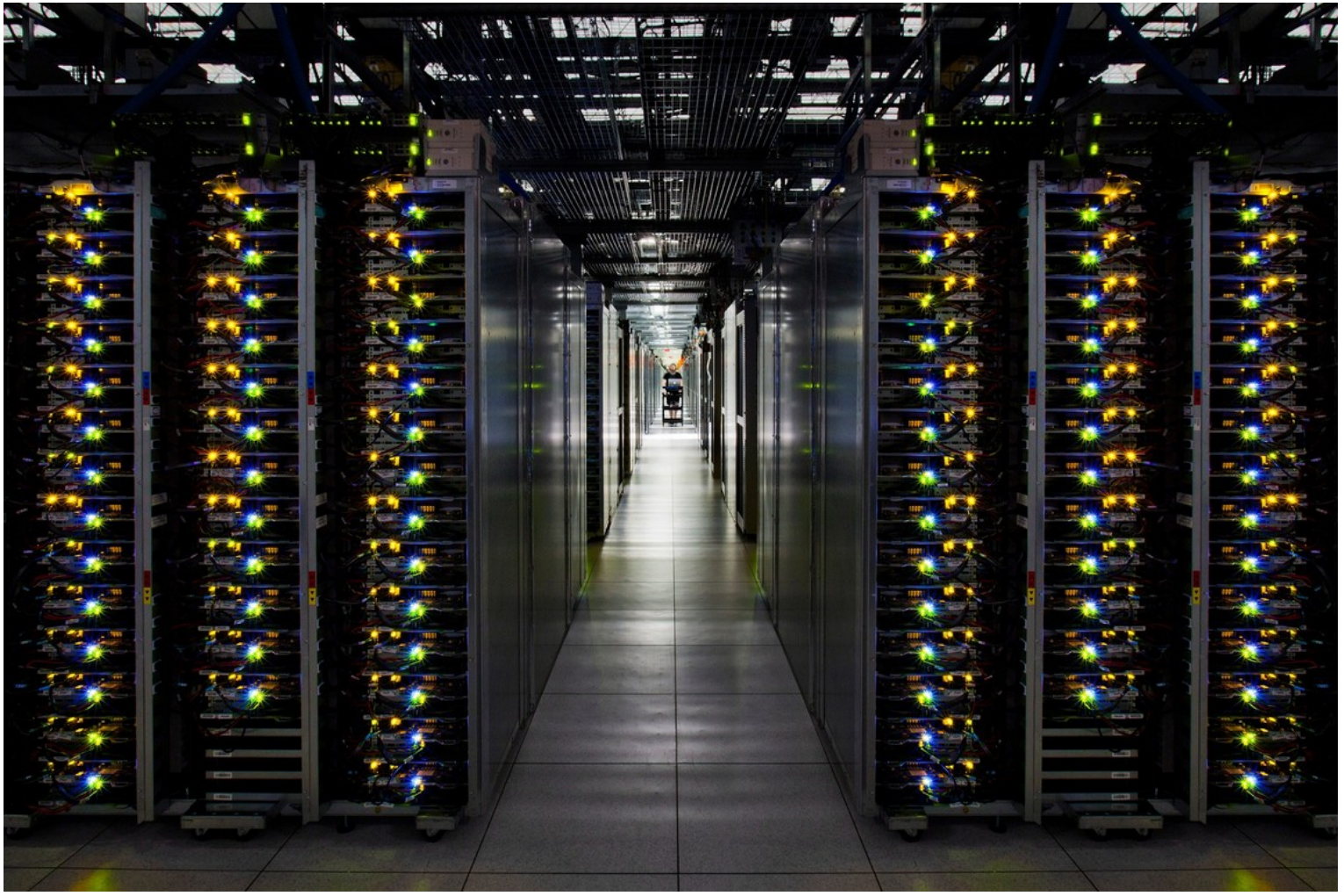
Como resolver problemas tão grandes?

- Estratégia simples (mas de difícil execução): Dividir para conquistar
- Usar mais recursos computacionais a medida que mais dados aparecerem

Grandes *data centers*

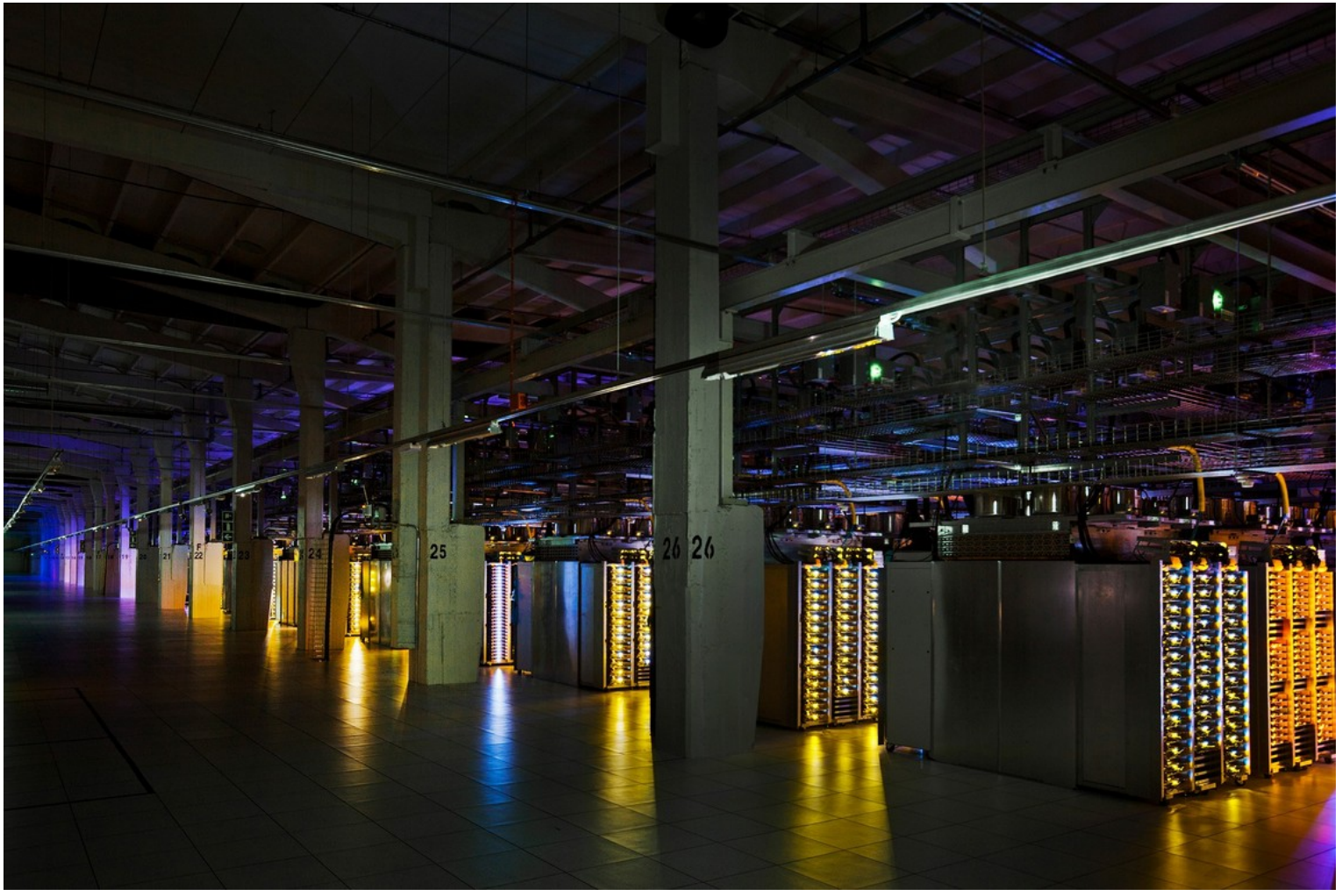
Pergunta:
Quão grandes são os *data centers* que fazem sistemas que afetam a vida de quase todo mundo que se conecta a Internet (como os do Google, Facebook, etc.) funcionarem?

Grandes *data centers*



Fonte: http://www.google.com/intl/pt-BR/about/datacenters/

Grandes *data centers*



Fonte: http://www.google.com/intl/pt-BR/about/datacenters/

Grandes *data centers*

Fonte: http://www.google.com/intl/pt-BR/about/datacenters/

Grandes *data centers*

Fonte: http://www.google.com/intl/pt-BR/about/datacenters/

Grandes *data centers*

Só o Google tem trezedesses espalhados pelo mundo!

Américas

- Berkeley County, Carolina do Sul Council
- Bluffs, Iowa
- Douglas County, Georgia Mayes County,
- Oklahoma Lenoir, Carolina do Norte The Dalles, Oregon Quilicura, Chile

Grandes *data centers*

Só o Google tem treze desses espalhados pelo mundo!

- Ásia
  - Hong Kong Cingapura Taiwan

Grandes *data centers*

Só o Google tem trezedesses espalhados pelo mundo!

- Europa
  - Hamina, Finlândia St Ghislain, Bélgica
    Dublin, Irlanda

Como isso era feito até então?

Evolução da computação

- anos 50: computadores eram grandes calculadoras programadas com cartões perfurados; início da computação paralela
- final dos anos 60: ARPANET (computadores começavam a serem interconectados; noção de computação *como um serviço*)
- anos 70: surgem os primeiros microprocessadores anos
- 80: popularização dos computadores pessoais anos 90:
- popularização da Internet

Paradigmas de computação

- Computadores Pessoais Computadores Paralelos
- Aglomerados de Computadores (*clusters*) Computação Voluntária (*Volunteer Computing*):
  - The Great Internet Mersenne Prime Search (1996): busca por primos de Mersenne (primos da forma $2^n - 1$, $n \in \mathrm{N}$)
  - distributed.net (1997): decriptografia por força-bruta
  - SETI@Home (1999): análise de sinais de rádio vindos do espaço em busca de evidência de vida extra-terreste
  - Computação em Grade (*Grid Computing*)

E como é feito agora?

Grandes data centers

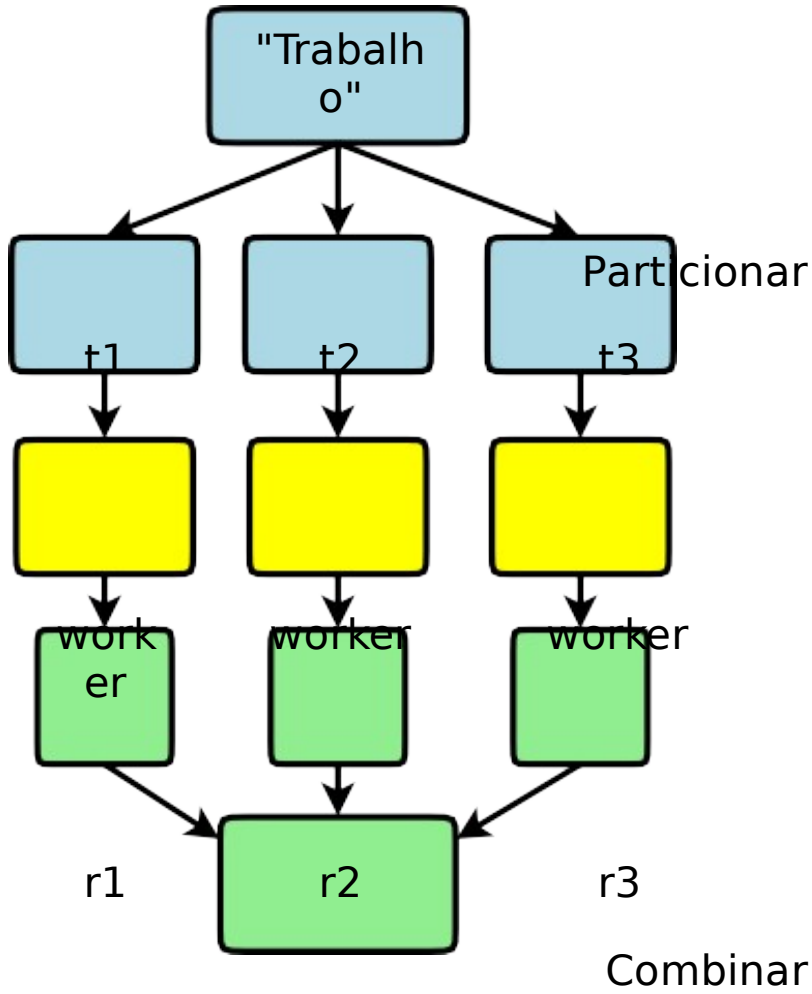
- Seu problema aumenta na mesma escala da web?
  Fácil: basta adicionar mais máquinas
  - Tendência: centralização dos recursos computacionais em grandes data centers
  - Problemas a serem resolvidos:
    Redundância Eficiência Utilização Gerenciamento

Ideias principais

- Escalabilidade horizontal, não vertical
  - Existem limites para máquinas SMP e arquiteturas de memória compartilhada
- Mova o processamento para perto dos dados
  - a banda de rede é limitada
- Processe os dados sequencialmente, evite padrões de acesso aleatórios
  - *seeks* são custosos, mas a vazão (*throughput*) do disco é razoável

Como programar aplicações escaláveis?

Divisão e conquista



Problema recorrente

- Problemas de paralelização surgem por causa de:
  - comunicação entre os *workers*
  - acesso a recursos compartilhados (por exemplo, dados)
- Portanto, precisamos de algum mecanismo de sincronização

Desafios de paralelização

- Como repartir as unidades de trabalho entre os *workers*?
- O que fazer quando temos mais trabalho do que *workers*?
- E se os *workers* precisarem compartilhar resultados intermediários entre si?
- Como agregar os resultados parciais?
- O que fazer se um *worker* parar de funcionar?
- Como saber se todos os *workers* terminaram seus trabalhos?

Desafios de paralelização

- Como repartir as unidades de trabalho entre os *workers*?
- O que fazer quando temos mais trabalho do que *workers*?
- E se os *workers* precisarem compartilhar resultados intermediários entre si?
- Como agregar os resultados parciais?
- O que fazer se um *worker* parar de funcionar?
- Como saber se todos os *workers* terminaram seus trabalhos?

Problema recorrente

- Problemas de paralelização surgem por causa de:
  - comunicação entre os *workers*
  - acesso a recursos compartilhados (por exemplo, dados)
- Portanto, precisamos de algum mecanismo de sincronização

Gerenciar múltiplos *workers*

- É difícil, pois:
  - Não sabemos em que ordem cada *worker* será executado
  - Não sabemos quando um *worker* irá interromper outro
  - *worker* Não sabemos em qual ordem os *workers* irão acessar os dados compartilhados
- Por tanto, nós precisamos de:
  - Semáforos (`lock, unlock`)
  - Variáveis condicionais (`wait, notify, broadcast`)
  - Barreiras de sincronização
- Ainda assim, restam problemas como:
  - Deadlock, starvation, race coditions, ...

Ferramentas atuais

- Modelos de programação:
  - Memória compartilhada (pthreads)
  - Passagem de mensagens (MPI)
- Padrões arquiteturais:
  - Mestre-escravo
  - Produtor-consumidor
- Filas de trabalho compartilhadas

Moral da história

- Tudo se resume ao nível mais adequado de abstração
- Esconda os detalhes do sistema dos desenvolvedores
  - Evita os problemas com race conditions, contenção em locks, etc.
- Separe o "quê" do "como":
  - O desenvolvedor especifica apenas o que deve ser computado
  - O arcabouço deve se encarregar de como realizar a execução

O data center *é* o computador!

Dúvidas