

Introdução à Modelagem e Execução de Workflows Científicos

Kelly R. Braghetto e Daniel Cordeiro
Departamento de Ciência da Computação
Instituto de Matemática e Estatística
Universidade de São Paulo

Jornadas de Atualização em Informática – julho de 2014

Olá! :)

Kelly Rosa Braghetto

é professora do Departamento de Ciência da Computação do IME-USP desde 2012. Possui mestrado (2006) e doutorado (2011) em Ciência da Computação pela Universidade de São Paulo. Desde 2003 desenvolve pesquisas na área e Modelagem de Dados e Processos, atuando principalmente nos seguintes temas: gerenciamento de workflows científicos e processos de negócio, modelagem formal de processos, avaliação de desempenho de processos via modelagem analítica e modelagem de bancos de dados.

Olá! :)

Daniel Cordeiro

é doutor em Mathématiques et en Informatique pela Université de Grenoble, França, e Mestre em Ciências e Bacharel em Ciência da Computação pela Universidade de São Paulo. Atualmente faz pós-doutorado em Ciência da Computação no IME-USP com financiamento da FAPESP. Há dez anos trabalha com pesquisa e desenvolvimento de aplicações computacionais de alto desempenho, utilizando técnicas variadas como análise de escalabilidade de servidores baseados em eventos, teoria do escalonamento, análise combinatória e teoria dos jogos.

Introdução

Os paradigmas da ciência

James Nicholas “Jim” Gray¹ dizia estarmos presenciando o começo de um **novo paradigma da ciência**.

- Há milhares de anos a ciência era **empírica**
- Há alguns séculos a ciência passou a ser também **teórica** (modelos, generalizações, etc.)
- Nas últimas décadas, cientistas passaram a validar seus modelos teóricos com o uso de **simulações**

¹Jim Gray ganhou o Prêmio Turing em 2008 e foi diretor do *Microsoft Research's eScience Group*.

Os paradigmas da ciência

James Nicholas “Jim” Gray¹ dizia estarmos presenciando o começo de um novo paradigma da ciência.

- Há milhares de anos a ciência era **empírica**
- Há alguns séculos a ciência passou a ser também **teórica** (modelos, generalizações, etc.)
- Nas últimas décadas, cientistas passaram a validar seus modelos teóricos com o uso de **simulações**

¹Jim Gray ganhou o Prêmio Turing em 2008 e foi diretor do *Microsoft Research's eScience Group*.

Os paradigmas da ciência

James Nicholas “Jim” Gray¹ dizia estarmos presenciando o começo de um novo paradigma da ciência.

- Há milhares de anos a ciência era **empírica**
- Há alguns séculos a ciência passou a ser também **teórica** (modelos, generalizações, etc.)
- Nas últimas décadas, cientistas passaram a validar seus modelos teóricos com o uso de **simulações**

¹Jim Gray ganhou o Prêmio Turing em 2008 e foi diretor do *Microsoft Research's eScience Group*.

Os paradigmas da ciência

James Nicholas “Jim” Gray¹ dizia estarmos presenciando o começo de um novo paradigma da ciência.

- Há milhares de anos a ciência era **empírica**
- Há alguns séculos a ciência passou a ser também **teórica** (modelos, generalizações, etc.)
- Nas últimas décadas, cientistas passaram a validar seus modelos teóricos com o uso de **simulações**

¹Jim Gray ganhou o Prêmio Turing em 2008 e foi diretor do *Microsoft Research's eScience Group*.

Um novo modo de se fazer ciência

O quarto paradigma seria a **exploração de dados**:

- Unifica teoria, experimentação e simulação
- “Toda ciência é ciência da computação”²
- Resultados de experimentos são expressos em

²George Johnson. “The World: In Silica Fertilization; All Science Is Computer Science”. Em: *The New York Times* (25 de mar. de 2001). URL: <http://www.nytimes.com/2001/03/25/weekinreview/the-world-in-silica-fertilization-all-science-is-computer-science.html>.

Um novo modo de se fazer ciência

O quarto paradigma seria a **exploração de dados**:

- Unifica teoria, experimentação e simulação
- “Toda ciência é ciência da computação”²
- Resultados de experimentos são expressos em

²George Johnson. “The World: In Silica Fertilization; All Science Is Computer Science”. Em: *The New York Times* (25 de mar. de 2001). URL: <http://www.nytimes.com/2001/03/25/weekinreview/the-world-in-silica-fertilization-all-science-is-computer-science.html>.

Um novo modo de se fazer ciência

O quarto paradigma seria a **exploração de dados**:

- Unifica teoria, experimentação e simulação
- “Toda ciência é ciência da computação”²
- Resultados de experimentos são expressos em

²George Johnson. “The World: In Silica Fertilization; All Science Is Computer Science”. Em: *The New York Times* (25 de mar. de 2001). URL: <http://www.nytimes.com/2001/03/25/weekinreview/the-world-in-silica-fertilization-all-science-is-computer-science.html>.

Um novo modo de se fazer ciência

O quarto paradigma seria a **exploração de dados**:

- Unifica teoria, experimentação e simulação
- “Toda ciência é ciência da computação”²
- Resultados de experimentos são expressos em bytes

²George Johnson. “The World: In Silica Fertilization; All Science Is Computer Science”. Em: *The New York Times* (25 de mar. de 2001). URL: <http://www.nytimes.com/2001/03/25/weekinreview/the-world-in-silica-fertilization-all-science-is-computer-science.html>.

Um novo modo de se fazer ciência

O quarto paradigma seria a **exploração de dados**:

- Unifica teoria, experimentação e simulação
- “Toda ciência é ciência da computação”²
- Resultados de experimentos são expressos em **muitos bytes**

²**George Johnson**. “The World: In Silica Fertilization; All Science Is Computer Science”. Em: *The New York Times* (25 de mar. de 2001). URL: <http://www.nytimes.com/2001/03/25/weekinreview/the-world-in-silica-fertilization-all-science-is-computer-science.html>.

Um novo modo de pensamento científico

A exploração de dados promove uma mudança importante no processo de pensamento científico.

Um novo modo de pensamento científico

A exploração de dados promove uma mudança importante no processo de pensamento científico.

Antes:

formulação de hipótese

Um novo modo de pensamento científico

A exploração de dados promove uma mudança importante no processo de pensamento científico.

Antes:

formulação de hipótese → experimentação

Um novo modo de pensamento científico

A exploração de dados promove uma mudança importante no processo de pensamento científico.

Antes:

formulação de hipótese → experimentação → análise de resultados

Um novo modo de pensamento científico

A exploração de dados promove uma mudança importante no processo de pensamento científico.

Antes:

formulação de hipótese → experimentação → análise de resultados

Agora: “*data-driven hypothesis*”

Um novo modo de pensamento científico

A exploração de dados promove uma mudança importante no processo de pensamento científico.

Antes:

formulação de hipótese → experimentação → análise de resultados

Agora: “*data-driven hypothesis*”

formulação de hipótese

Um novo modo de pensamento científico

A exploração de dados promove uma mudança importante no processo de pensamento científico.

Antes:

formulação de hipótese → experimentação → análise de resultados

Agora: “*data-driven hypothesis*”

formulação de hipótese → busca da confirmação no banco de dados

Um novo modo de pensamento científico

A exploração de dados promove uma mudança importante no processo de pensamento científico.

Antes:

formulação de hipótese → experimentação → análise de resultados

Agora: “*data-driven hypothesis*”

formulação de hipótese → busca da confirmação no banco de dados

Trata-se realmente de uma nova metodologia

que permite a criação de novos “tipos” de pesquisa em diversas áreas do conhecimento (notadamente nas ciências naturais).

O 4º paradigma: exploração dos dados

- 1 Os dados são capturados por instrumentos ou gerados por simulações
- 2 Processados por sistemas de software complexos
- 3 A informação (ou conhecimento) resultante é armazenada no computador
- 4 Só então os cientistas analisam os dados, no final do processo! (usando ferramentas de gerenciamento de dados e de estatística)

Exemplo

Hoje podemos dizer que astrônomos não olham mais através de seus telescópios. Ao invés disso, eles “olham” através de instrumentos complexos que estão conectados a centrais de processamento de dados e, só então, utilizam seus computadores para visualizar as informações coletadas.

O 4º paradigma: exploração dos dados

- 1 Os dados são capturados por instrumentos ou gerados por simulações
- 2 Processados por sistemas de software complexos
- 3 A informação (ou conhecimento) resultante é armazenada no computador
- 4 Só então os cientistas analisam os dados, no final do processo! (usando ferramentas de gerenciamento de dados e de estatística)

Exemplo

Hoje podemos dizer que astrônomos não olham mais através de seus telescópios. Ao invés disso, eles “olham” através de instrumentos complexos que estão conectados a centrais de processamento de dados e, só então, utilizam seus computadores para visualizar as informações coletadas.

O 4º paradigma: exploração dos dados

- 1 Os dados são capturados por instrumentos ou gerados por simulações
- 2 Processados por sistemas de software complexos
- 3 A informação (ou conhecimento) resultante é armazenada no computador
- 4 Só então os cientistas analisam os dados, no final do processo! (usando ferramentas de gerenciamento de dados e de estatística)

Exemplo

Hoje podemos dizer que astrônomos não olham mais através de seus telescópios. Ao invés disso, eles “olham” através de instrumentos complexos que estão conectados a centrais de processamento de dados e, só então, utilizam seus computadores para visualizar as informações coletadas.

O 4º paradigma: exploração dos dados

- 1 Os dados são capturados por instrumentos ou gerados por simulações
- 2 Processados por sistemas de software complexos
- 3 A informação (ou conhecimento) resultante é armazenada no computador
- 4 Só então os cientistas analisam os dados, **no final do processo!** (usando ferramentas de gerenciamento de dados e de estatística)

Exemplo

Hoje podemos dizer que astrônomos não olham mais através de seus telescópios. Ao invés disso, eles “olham” através de instrumentos complexos que estão conectados a centrais de processamento de dados e, só então, utilizam seus computadores para visualizar as informações coletadas.

O 4º paradigma: exploração dos dados

- 1 Os dados são capturados por instrumentos ou gerados por simulações
- 2 Processados por sistemas de software complexos
- 3 A informação (ou conhecimento) resultante é armazenada no computador
- 4 Só então os cientistas analisam os dados, **no final do processo!** (usando ferramentas de gerenciamento de dados e de estatística)

Exemplo

Hoje podemos dizer que astrônomos não olham mais através de seus telescópios. Ao invés disso, eles “olham” através de instrumentos complexos que estão conectados a centrais de processamento de dados e, só então, utilizam seus computadores para visualizar as informações coletadas.

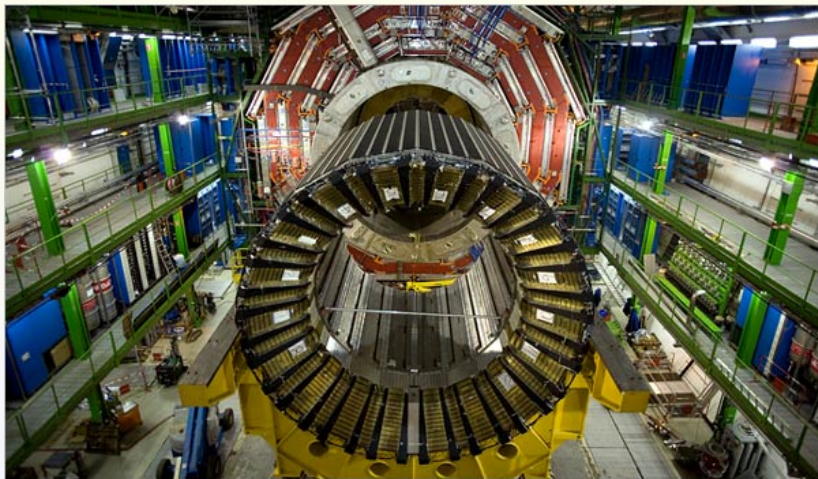
De qual volume de dados estamos falando?

Problemas da ordem de **petabytes**!

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

- └ O quarto paradigma
- └ Dilúvio de dados

De qual volume de dados estamos falando?



De qual volume de dados estamos falando?



De qual volume de dados estamos falando?

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

Large Hadron Collider (LHC)

Produce cerca de **25 petabytes/ano**, processa cerca de **um petabyte de dados todos os dias** (o equivalente a cerca de 210.000 DVDs) e já possui mais de **100 petabytes armazenados** desde o início do projeto ($\cong$ 700 anos de filmes armazenados com qualidade *full HD*)

De qual volume de dados estamos falando?

Outros exemplos comerciais

- 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)

De qual volume de dados estamos falando?

Outros exemplos comerciais

- 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)

De qual volume de dados estamos falando?

Outros exemplos comerciais

- 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)

De qual volume de dados estamos falando?

Outros exemplos comerciais

- 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)

Grandes *data centers*

Pergunta:

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

- └ O quarto paradigma
- └ Dilúvio de dados

Grandes *data centers*



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

- └ O quarto paradigma
- └ Dilúvio de dados

Grandes *data centers*



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

- └ O quarto paradigma
- └ Dilúvio de dados

Grandes *data centers*



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

- └ O quarto paradigma
- └ Dilúvio de dados

Grandes *data centers*



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

Difícilmente alguém consegue processar tudo isso sozinho

- Nem todos podem comprar supercomputadores
- Nem todos podem arcar com os custos de manter um *data center* grande e potente o suficiente (custos de operação, manutenção, etc.)
- O uso de Grades Computacionais (a solução mais usada até o momento) também depende de acesso a aglomerados de computação e a supercomputadores (mesmo que eles pertençam a outros) e requer que os participantes estejam em comum acordo

Difícilmente alguém consegue processar tudo isso sozinho

E ainda que tenhamos tudo isso à disposição

A grande pergunta é:

Como modelar programas e experimentos que possam usar todo esse poder computacional de forma simples, barata e eficiente?

Workflows Científicos

Definição

“Redes de processos tipicamente utilizadas como ‘pipelines de análises de dados’ ou ainda para comparar dados observados ou previstos, e que podem incluir uma vasta gama de componentes, e.g. para consultar bancos de dados, para transformar ou minerar dados, para executar simulações em computadores de alto desempenho, etc.”³

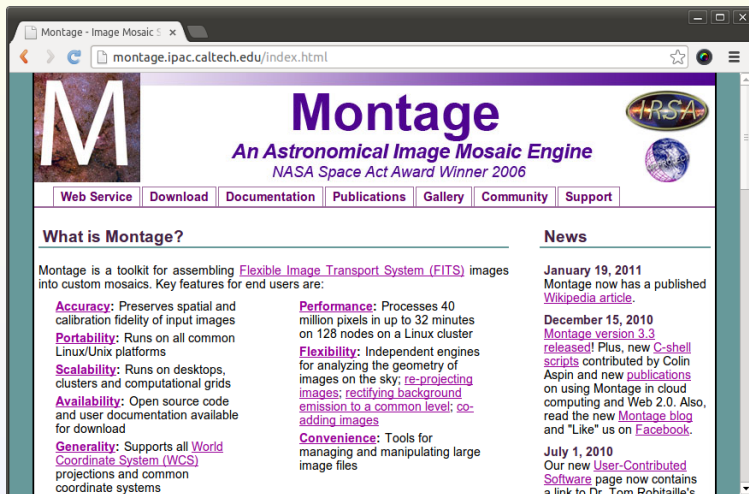
³Bertram Ludäscher et al. “Scientific workflow management and the Kepler system”. Em: *Concurrency and Computation: Practice and Experience* 18.10 (2006), pp. 1039–1065.

Workflows Científicos

- São automações de experimentos ou de processos científicos, expressas em termos das atividades a serem executadas e, principalmente, das dependências dos dados manipulados;
- Tendem a ser computacionalmente intensivos;
- São fortemente voltados à manipulação de grandes volumes de dados complexos.

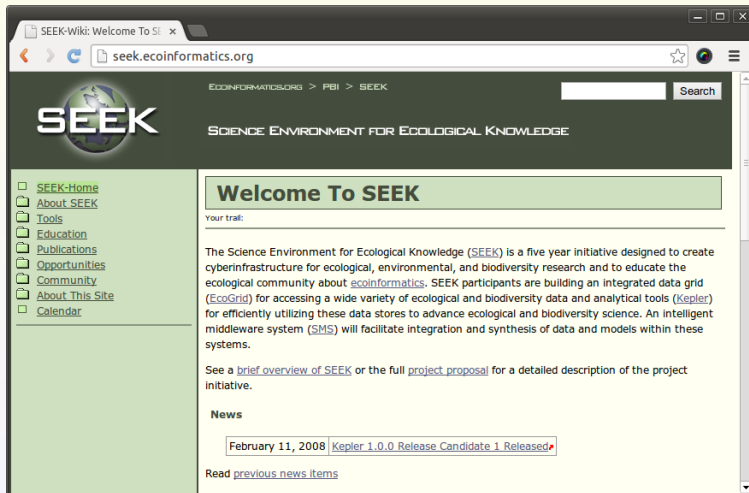
- └ O quarto paradigma
- └ Workflows científicos

Exemplos famosos

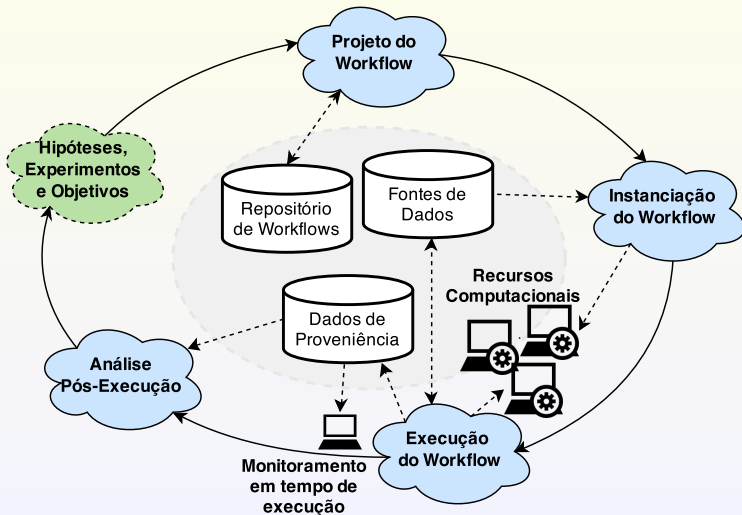


- ↳ O quarto paradigma
- ↳ Workflows científicos

Exemplos famosos



Ciclo de vida dos workflows científicos



O projeto de workflows científicos

- Consiste na construção de um modelo para o workflow
- Pode ser feito por meio de diferentes linguagens ou modelos de representação
- Pode se embasar em outros workflows já existentes
- É conveniente de se representar em formatos digitais e armazenar em repositórios:
 - reuso, compartilhamento com outros pesquisadores, etc.

A instanciação de workflows científicos

- Corresponde à preparação necessária para uma execução particular do workflow:
 - Cada nova execução requer dados de entrada e parâmetros de configuração definidos pelos cientistas
- Pode envolver a seleção e alocação dos recursos computacionais para a execução
- Cuida da transferência dos dados de entrada para os computadores onde a execução será realizada

A execução de workflows científicos

- Ocorre nos recursos computacionais disponíveis:
 - Para cada atividade atribui-se um recurso
 - Atividades processam dados de entrada e produzem novos dados, que podem ser usados por outras atividades
- Pode registrar dados intermediários gerados na instância e suas *informações de proveniência*
 - Dados de entrada e os parâmetros de configuração
 - Registro das atividades já executadas, seus tempos de início e término, os recursos usados, etc.
 - Referências para os dados de entrada e saída de cada atividade

Intenções: avaliação dos resultados; recuperação em caso de falhas; reprodutibilidade dos experimentos

A análise pós-execução de workflows científicos

- Corresponde à inspeção e interpretação dos resultados obtidos
- Pode conduzir a uma revisão da hipótese ou do objetivo experimental inicial, ou ainda à identificação de ineficiências da execução
 - Reprojetar o workflow $\Rightarrow$ nova iteração do seu ciclo de vida

Modelagem de Workflows Científicos

Terminologia básica sobre representação de workflows

- Modelo** define as atividades a serem realizadas no workflow para que um objetivo científico seja alcançado
- Atividade** é uma unidade atômica (= indivisível) de trabalho em um modelo de workflow
- Conector** é usado para definir alguma relação de precedência ou algum tipo de transferência de dados entre atividades
- Instância** é uma execução particular (= instanciação) de um modelo de workflow

Modelo de workflow

Define as atividades a serem realizadas no workflow para que um objetivo científico seja alcançado

- Estabelece a ordem (ainda que parcial) na qual as atividades do workflow devem ser executadas
- É comumente representado por meio de um **arcabouço formal** ou de uma **linguagem de especificação de workflows**
 - arcabouços formais possibilitam a análise de propriedades dos workflows que neles são modelados
 - linguagens específicas de domínio permitem a especificação de modelos em um nível de detalhamento mais próximo ao requerido para a execução dos workflows

Atividade

Unidade atômica (= indivisível) de trabalho em um modelo de workflow

- Possui vários sinônimos: *tarefa*, *processo*, *ação*, *transição*
- Pode ser classificada como:
 - Atividade manual** quando é realizada por pessoas
 - Atividade automática** quando é realizada completamente por computadores ou outros tipos de máquinas, sem intervenção humana
 - Atividade semiautomática** quando depende tanto de pessoas quanto de máquinas para ser realizada

Conector

Define alguma relação de precedência ou algum tipo de transferência de dados entre atividades

- Cada linguagem de modelagem de workflows possui o seu próprio conjunto de conectores
- Os conectores definem a expressividade da linguagem, ou seja, a **variedade** e **quantidade** de modelos de workflows que a linguagem é capaz de representar

Instância de workflow

Execução particular (= instanciação) de um modelo de workflow

- Duas instâncias de um mesmo workflow não necessariamente executam a mesma sequência de atividades
 - Um workflow pode ter em seu modelo ramos de execução alternativos ou atividades que só serão executadas caso determinados tipos de dados estejam disponíveis

Workflows de Negócio × Workflows Científicos

Workflows de Negócio

- São usados em grande escala desde os anos 90
- Automatizam processos industriais e gerenciais (= *processos de negócio*)
- Desenvolvimento apoiado por comunidades e coalizões, envolvendo indústria e academia

Resultados:

- Muitos documentos, modelos de referência e padrões relacionados à gestão desse tipo de workflows
- Muitas técnicas e ferramentas computacionais desenvolvidas para apoiar o ciclo de vida desses workflows

Workflows de Negócio × Workflows Científicos

Workflows Científicos

- Sua popularização começou nos anos 2000
- Possuem muitas semelhanças com os workflows de negócio
- A comunidade científica não adotou os padrões desenvolvidos para os workflows de negócios
 - O domínio científico possui particularidades que justificam a criação de técnicas e ferramentas específicas para o gerenciamento de seus workflows

Padronização e Interoperabilidade

Workflows de negócio

- Processos de negócio envolvem processos inter-organizacionais
- Demandam interoperabilidade de ferramentas e padronização das linguagens de modelagem
 - Exemplos: o modelo de referência da *Workflow Management Coalition* e a *Business Process Model and Notation* (BPMN)

Workflows científicos

- Não existem padrões: cada sistema de gerenciamento possui a sua linguagem de modelagem
- Pouca interoperabilidade entre diferentes ferramentas
 - Motivo: poucos domínios científicos (como a Bioinformática) fazem uso sistematizado de ferramentas de workflows

Padronização e Interoperabilidade

- Workflows de negócio são projetados e desenvolvidos por especialistas em TI
 - Atividades são componentes padronizados e portáteis (como os serviços Web)
 - Existe facilidade no reúso e na construção dos processos inter-organizacionais
- Workflows científicos são projetados por cientistas
 - Atividades muitas vezes são *scripts* para softwares matemáticos ou estatísticos
 - Outras vezes, o próprio workflow é implementado como um *script* que coordena as chamadas a outros *scripts*

A proposição de padrões para a gestão de workflows no domínio científico é uma tarefa desafiadora!

Natureza das atividades

- Workflows de negócio comumente possuem atividades automáticas e manuais
 - Exemplo: um workflow para o tratamento de reclamações de clientes de uma empresa de TV a cabo
- Workflows científicos não costumam envolver atividades manuais pois suas ferramentas de apoio não oferecem suporte a esse tipo de funcionalidade

Integridade na execução de instâncias

Workflows de negócio

- Integridade na execução é uma preocupação crucial
- Podem implementar transações comerciais que envolvem contratos financeiros e prestação de serviços
- Geralmente, são workflows transacionais
 - Em caso de falhas durante sua execução, é preciso haver maneiras de se desfazer as atividades já realizadas (mesmo que seja de maneira lógica)

Integridade na execução de instâncias

Workflows científicos

- Quando têm como alvo o processamento e a análise de dados, geralmente não demandam tratamento transacional
- No caso de falhas durante a execução, remove-se os resultados intermediários produzidos e então reinicia-se a execução do workflow a partir do ponto onde ela foi interrompida ou do seu início
 - A reexecução pode ter um custo alto em termos de tempo e de recursos computacionais consumidos

Perspectivas para a modelagem de workflows

Fluxo de controle descreve a ordem de execução, definindo a transferência de controle entre atividades conectadas e habilitando uma atividade para execução quando as atividades que a precedem são concluídas

Fluxo de dados descreve as transferências de dados entre as atividades, estabelecendo as dependências das atividades com relação aos dados que elas manipulam

Organizacional atrela ao workflow uma estrutura organizacional, por meio da definição de papéis (desempenhados por pessoas ou equipamentos) responsáveis pela execução das atividades

Tratamento de exceções lida com as causas das exceções e as ações que precisam ser tomadas nos seus tratamentos

Perspectivas para a modelagem de workflows

Fluxos de dados

- Em workflows científicos, a predominância é da modelagem de fluxos de dados
 - Aplicações científicas são intensivas em dados
- Cientistas rotineiramente já descrevem seus experimentos científicos em termos de fluxos de dados
 - coleta de dados brutos → filtragem → transformação → análise → obtenção de dados derivados que sustentam hipóteses científicas

Perspectivas para a modelagem de workflows

Modelagem híbrida

- As diferentes perspectivas de modelagem não são excludentes
- Muitos SGWC possuem linguagens de modelagem híbridas, para a descrição de fluxos tanto de dados quanto de controle
- Ferramentas de gestão de workflows de negócio costumam combinar as perspectivas de fluxo de controle e de tratamento de exceções

Estruturas para representação de fluxos de controle

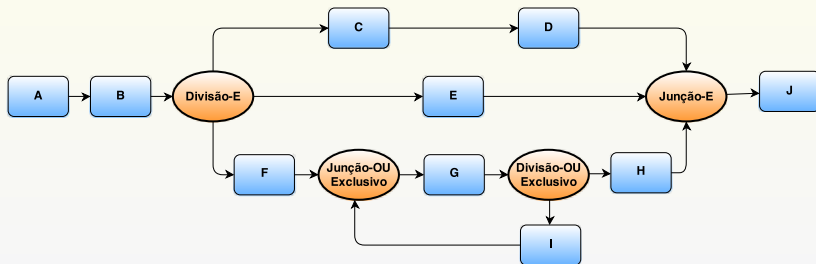
Construções essenciais:

- *Sequência*
- *Paralelismo (Divisão-E)*
- *Sincronização (Junção-E)*
- *Escolha Exclusiva (Divisão-OU-Exclusivo)*
- *Junção (Junção-OU-Exclusivo)*
- *Ciclo (ou Iteração)*

Outras construções

- A *Workflow Patterns Initiative* identificou as construções mais frequentes nas ferramentas de modelagem:
 - há 43 padrões diferentes para fluxo de controle

Estruturas para representação de fluxos de controle



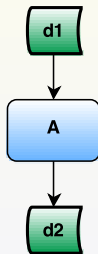
Exemplo de workflow modelado usando as construções básicas de fluxo de controle

Estruturas para representação de fluxos de dados

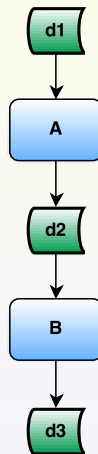
Construções básicas:

- **Processamento**: é realizado por uma atividade sobre dados de entrada para produzir dados de saída
- **Pipeline**: combina um ou mais processamentos sequencialmente, de forma que cada atividade processe os dados produzidos pela atividade que a precedeu e que a sua saída seja usada como entrada para a próxima atividade

Estruturas para representação de fluxos de dados



Processamento



Pipeline

Estruturas para representação de fluxos de dados

Construções básicas:

- ***Distribuição de dados***: é feita por uma atividade que produz dois ou mais conjuntos de dados de saída que são recebidos como entrada por duas ou mais atividades

Pode ser feita para dois propósitos distintos:

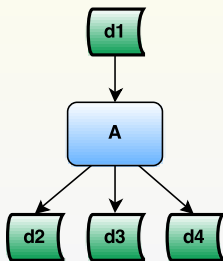
- produzir novos conjuntos de dados para serem consumidos por múltiplas atividades; ou
- particionar um grande conjunto de dados recebido como entrada em subconjuntos menores, para que esses possam ser processados (paralelamente) por outras atividades no workflow

Estruturas para representação de fluxos de dados

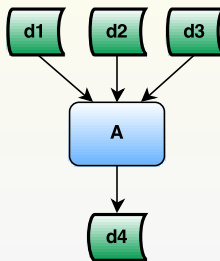
Construções básicas:

- **Agregação de dados:** é feita por uma atividade que agrega e processa dados de saída de duas ou mais atividades, gerando uma combinação dos dados como saída
 - Pode resultar em sincronização de linhas de execução paralelas
- **Redistribuição de dados:** é feita por uma atividade que combina a função de agregação de dados à função de distribuição dos dados
 - a atividade pode tanto sincronizar linhas de execução paralelas já existentes quanto criar novas linhas de execução paralelas

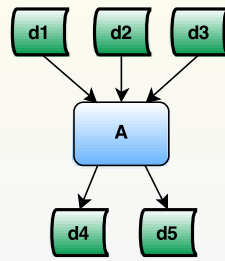
Estruturas para representação de fluxos de dados



Distribuição
de dados

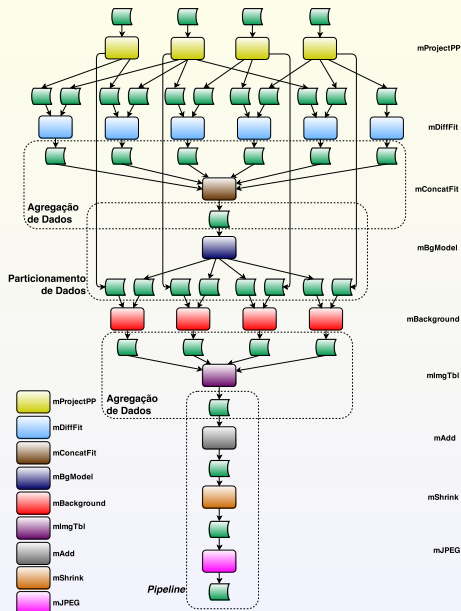


Agregação
de dados

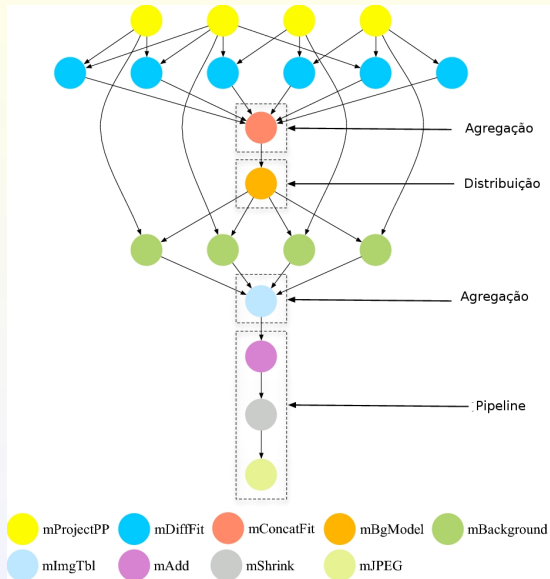


Redistribuição
de dados

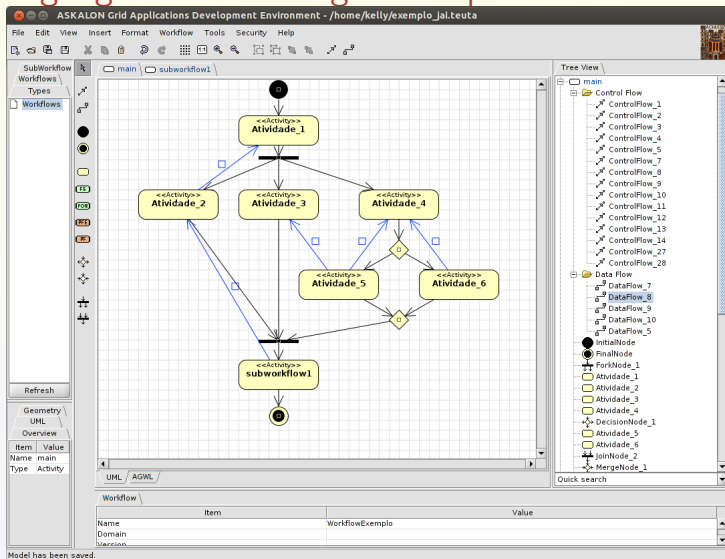
Fluxo de dados
no workflow
Montage
[http://montage.
ipac.caltech.edu/](http://montage.ipac.caltech.edu/)



Workflow Montage (sem a visualização dos itens de dados)

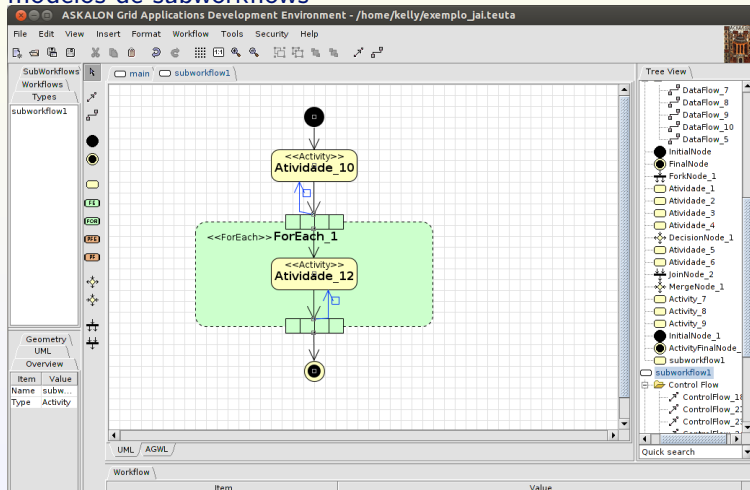


As linguagens de modelagem “na prática”



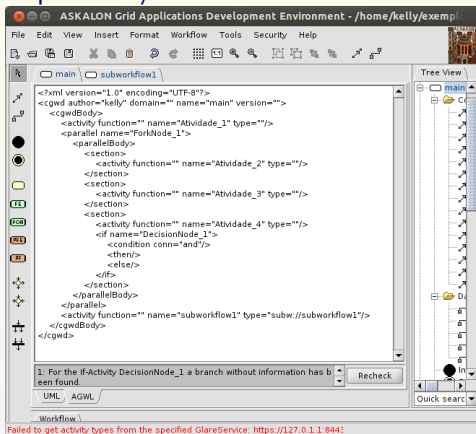
As linguagens de modelagem “na prática”

Composicionalidade: construção de modelos de workflow a partir de modelos de subworkflows



As linguagens de modelagem “na prática”

Representação textual do modelo



Modelagem formal de workflows

Por que usá-la?

- Geralmente, a semântica operacional dos conectores e outros construtores presentes nas linguagens de modelagem de workflows não é definida formalmente
 - Isso impossibilita a verificação e a validação dos modelos
 - A partir de modelos formais, é possível fazer análises preditivas qualitativas e quantitativas dos workflows

Classes de formalismos mais usados na modelagem de workflows:

- Redes de Petri (RdPs)
- Álgebras de Processos

Breve introdução às Redes de Petri (RdPs)

- A teoria das Redes de Petri é um dos exemplos mais conhecidos de teoria de ordem parcial para modelagem e análise de sistemas concorrentes
- O conceito foi introduzido por Carl Adam Petri, em sua tese de doutorado em 1962, na Faculdade de Matemática e Física da Universidade de Darmstadt, Alemanha
- São muito utilizadas devido à sua representação gráfica de fácil compreensão e ao seu potencial matemático para a análise de modelos
 - Essas análises incluem verificações de propriedades inerentes aos sistemas concorrentes, como relações de precedência entre eventos, sincronização e existência ou não de impasses

Elementos de uma RdP

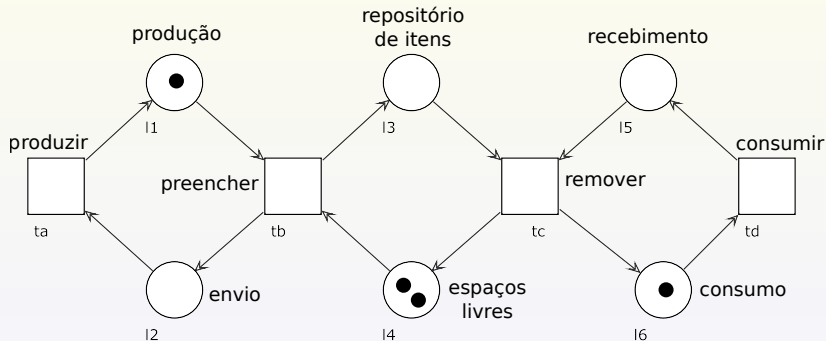
As RdPs são formadas basicamente por dois tipos de componentes

- **Transições** representando ações do sistema
- **Lugares** representando variáveis de estado do sistema

Uma RdP com marcação é formalmente definida pela tupla $RdP = \{L, T, A, P, m_0\}$, em que:

- $L = \{l_1, l_2, \dots, l_L\}$ é um conjunto de lugares;
- $T = \{t_1, t_2, \dots, t_T\}$ é um conjunto de transições;
- $A \subseteq (L \times T) \cup (T \times L)$ é o conjunto de arcos;
- $P : A \rightarrow \mathbb{N}$ é a função peso dos arcos;
- $m_0 = \{m_{01}, m_{02}, \dots, m_{0L}\}$ é a marcação inicial da rede.

Exemplo de RdP: Processos Produtor-Consumidor



Elementos de uma RdP

Os conceitos associados a cada um dos itens de uma RdP são:

- **Lugar** (representado graficamente por um círculo)
 - Modela uma condição que deve ser satisfeita para que o disparo da transição seja realizado
- **Transição** (representada graficamente por um quadrado, retângulo ou barra)
 - Pode ser compreendida como uma ação ou evento
- **Arco orientado**
 - Liga um lugar a uma transição ou vice-versa, encadeando condições e eventos

Elementos de uma RdP

Os conceitos associados a cada um dos itens de uma RdP são:

- **Marca ou Ficha** – representa um recurso disponível
 - O posicionamento dessas fichas nos lugares do grafo constitui a **marcação** da RdP
 - Cada lugar possui 0 ou mais fichas
 - A evolução da marcação permite modelar o comportamento dinâmico do sistema
- **Peso** – cada arco possui um peso associado a ele
 - O peso indica quantas fichas uma transição consome de um lugar de entrada ou quantas fichas uma transição acrescenta em um lugar de saída
 - Quando um arco não possui um peso explicitamente indicado no grafo, considera-se que o seu peso é 1

Marcação de uma RdP

- Uma marcação é uma atribuição de fichas a lugares
- Pode ser representada por um vetor cujo tamanho é o número de lugares na rede: o $i^{\text{ésimo}}$ componente do vetor representa o número de fichas contidas no lugar l_i
- Denotamos por $m(l)$ o número de fichas em um lugar $l \in L$

Evolução da marcação de uma RdP

- A evolução dinâmica da marcação de uma RdP é governada pela ocorrência (disparos) de transições que consomem e produzem fichas
- Regras de **habilitação** e **disparo** estão associadas às transições

Habilitação de uma transição em uma RdP

■ Habilitação

Um transição t está habilitada na marcação m se e somente se:

$$\forall l \in \bullet t, \quad m(l) \geq P(\langle l, t \rangle)$$

onde $\bullet t = \{u \mid \langle u, t \rangle \in A\}$.

Em outras palavras, $\bullet t$ é o conjunto de todos os lugares de entrada de t .

Disparo de uma transição em uma RdP

■ Disparo

O disparo de uma transição t , habilitada na marcação m , produz um marcação m' tal que

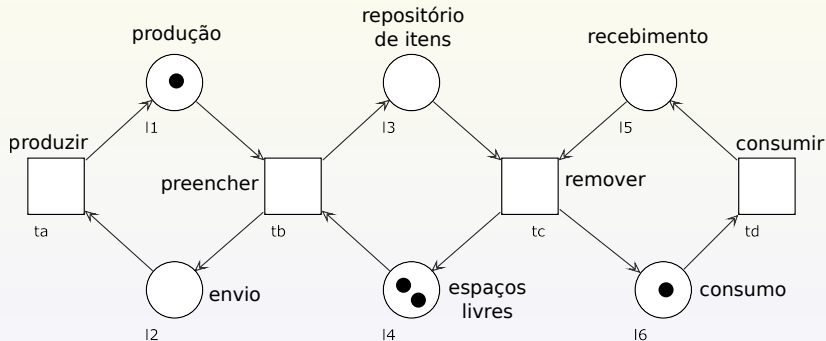
$$m' = m + O(t) - I(t) ,$$

em que

$$I(t) = (P(\langle l_1, t \rangle), P(\langle l_2, t \rangle), \dots, P(\langle l_L, t \rangle)) \quad \text{e} \\ O(t) = (P(\langle t, l_1 \rangle), P(\langle t, l_2 \rangle), \dots, P(\langle t, l_L \rangle))$$

- Essa declaração é normalmente indicada de forma compacta como $m[t]m'$ e podemos dizer que m' é diretamente alcançável a partir de m

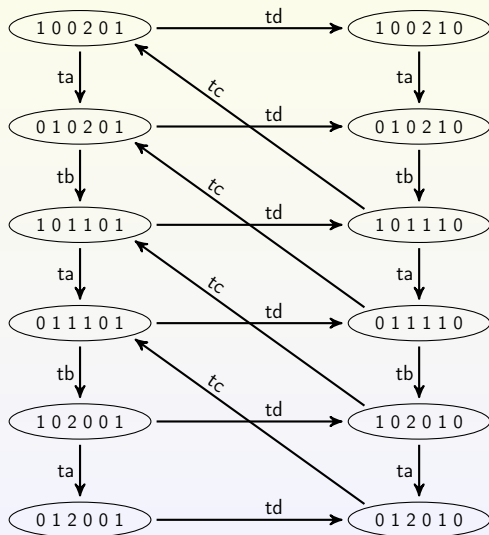
Exemplo de RdP: Processos Produtor-Consumidor



Exemplo de RdP: Processos Produtor-Consumidor

- O produtor só produz um item por vez, ao passo que o consumidor só consome um item por vez
- O produtor só pode enviar um item produzido a um consumidor quando há espaço para armazená-lo, ou seja, quando há ao menos uma ficha em l_4
- O consumidor só pode consumir um item após a sua remoção do repositório (transição t_c). Essa remoção só pode ocorrer quando há itens produzidos e ainda não consumidos, ou seja, quando há ao menos uma ficha em l_3
- Sendo assim, uma ficha que sai de l_4 com o disparo de t_b entra em l_3
- Do mesmo modo, uma ficha que sai de l_3 com o disparo de t_c entra em l_4

Marcações alcançáveis da rede do exemplo



Sobre as marcações de uma RdP

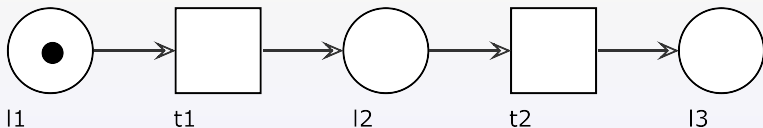
- Cada uma das marcações possíveis da rede representa um possível estado do sistema modelado
- A partir dos estados e da estrutura da rede, é possível verificar propriedades do sistema. Exemplos:
 - Se existe uma transição que não é habilitada por nenhuma marcação da rede, então essa transição é *morta* (ou seja, ela nunca será disparada) $\Rightarrow$ isso pode indicar um erro no projeto do sistema
 - Pode-se querer saber se o sistema pode atingir um determinado estado indesejado. Para isso, identifica-se qual é a marcação da RdP que representa esse estado e caso essa marcação esteja entre as alcançáveis da rede, então estará confirmado que o estado indesejado pode ocorrer no sistema

Mapeamento das estruturas básicas de workflows para RdPs

Sequência ou *Pipeline*

Quando duas atividade devem ser executadas sequencialmente, ou quando uma produz dados que são usados pela outra, então há uma interdependência entre elas

- Em RdPs, isso pode ser modelado conectando as duas atividades (transições) por meio de um lugar



Mapeamento das estruturas básicas de workflows para RdPs

Paralelismo ou distribuição de dados

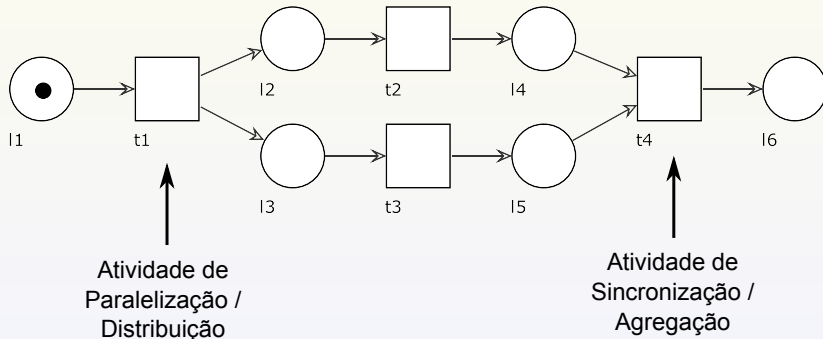
- Modelado por uma atividade (transição) que, quando disparada, inclui uma ficha em cada um dos lugares de entrada das transições que representam as atividades que serão executadas paralelamente ou que processarão os dados distribuídos

Sincronização ou agregação de dados

- É modelada por uma atividade que tem como lugar de entrada os lugares de saída que indicam quando as atividades a serem sincronizadas já foram concluídas ou quando os dados a serem agregados já estão disponíveis para processamento

Mapeamento das estruturas básicas de workflows para RdPs

Paralelismo e sincronização / Distribuição e agregação de dados



Mapeamento das estruturas básicas de workflows para RdPs

Escolha exclusiva

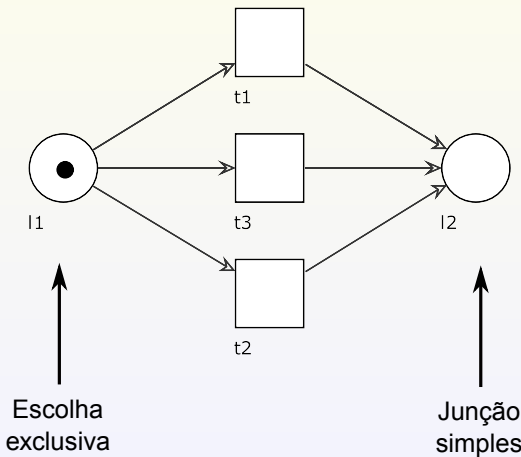
- Pode ser representada como um conjunto de atividades (transições) compartilhando um único lugar de entrada
- Há variações possíveis nessa representação
 - Exemplo: a escolha pode depender de uma regra que pode ser modelada como uma atividade “gerencial”, que quando disparada coloca uma ficha no lugar de entrada da atividade escolhida para execução

Junção simples

- É modelada por um conjunto de atividades (transições) alimentando um mesmo lugar de saída

Mapeamento das estruturas básicas de workflows para RdPs

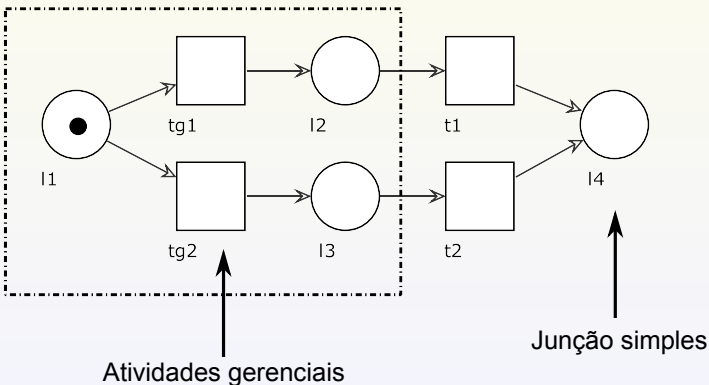
Escolha exclusiva e junção simples



Mapeamento das estruturas básicas de workflows para RdPs

Escolha exclusiva baseada em regras e junção simples

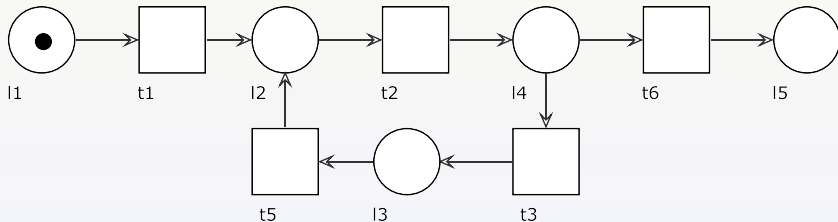
Escolha exclusiva



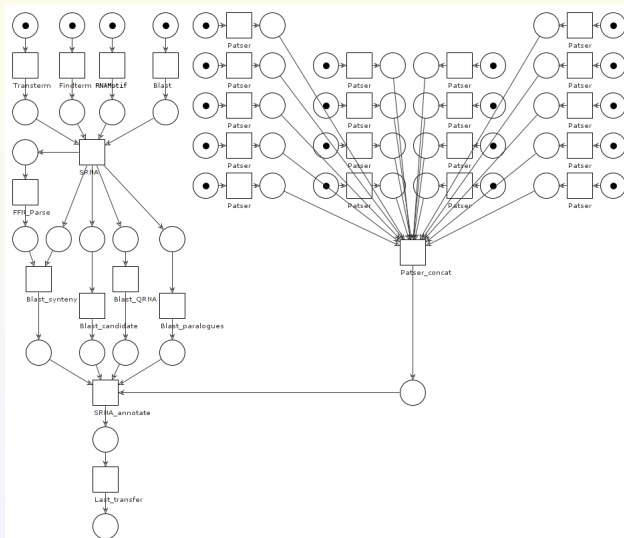
Mapeamento das estruturas básicas de workflows para RdPs

Ciclos

- Geralmente são modelados com o auxílio de escolhas exclusivas e junções simples



O workflow científico SIPHT modelado em RdP



Brevíssima introdução às Álgebras de Processos

- Nas *álgebras de processos* (APs) os processos são representados **algebricamente**, na forma de termos
- Uma AP é composta por um conjunto de **símbolos de ações**, um conjunto de **operadores** e um conjunto de **axiomas** descrevendo as propriedades dos operadores
- O conjunto de axiomas (= leis equacionais) pode também especificar quando dois processos são considerados equivalentes
- Esse arcabouço algébrico possibilita o raciocínio formal sobre **propriedades estruturais e comportamentais** de um modelo de sistema concorrente

APs – Origens

- As bases das álgebras de processos foram desenvolvidas, independentemente, por Milner e Hoare (no anos 70)
- As APs são parcialmente enraizadas nas Redes de Petri, na Teoria de Autômatos e nas linguagens formais
- Milner desenvolveu a álgebra de processos ***Calculus of Communicating Systems*** (CCS), enquanto Hoare definiu a ***Communicating Sequential Processes***(CSP)
- Outras álgebras de processos bem difundidas são a ***Language of Temporal Ordering Specification*** (LOTOS), a ***Algebra of Communicating Processes*** (ACP) e a ***π -Calculus***

Assinatura de uma AP básica

Geralmente consiste em:

- um conjunto finito e não vazio A de **ações atômicas**, representando comportamentos indivisíveis
- um operador binário “+”, chamado de **composição alternativa**

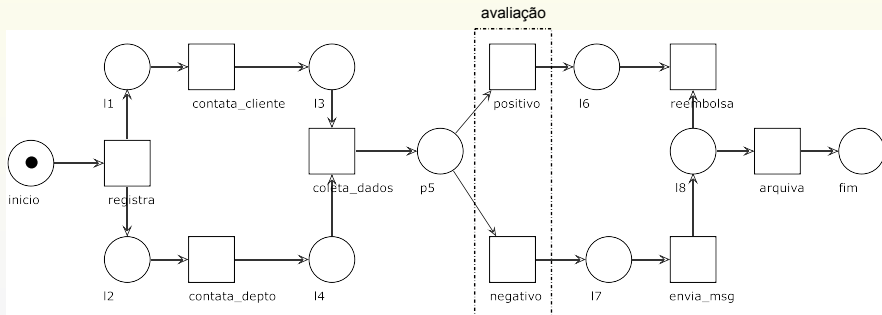
Se os termos fechados t_1 e t_2 representam os processos p_1 e p_2 , respectivamente, então o termo fechado $t_1 + t_2$ representa o processo que executa p_1 ou p_2

- um operador binário “.”, chamado de **composição sequencial**
Se os termos fechados t_1 e t_2 representam os processos p_1 e p_2 , respectivamente, então o termo fechado $t_1.t_2$ representa o processo que executa primeiro p_1 e depois p_2

Outros operadores das APs

- Para modelar cenários que envolvem **paralelismo**, **comunicação** e **iterações**, as álgebras de processos geralmente possuem, além dos operadores básicos, mecanismos para expressar comportamentos mais avançados
- Por exemplo, na *Algebra of Communicating Processes* (ACP) existe:
 - o operador binário “ $\parallel$ ”, chamado de **entrelaçamento**, para expressar paralelismo
 - **equações recursivas**, para representação de ciclos finitos ou infinitos

Exemplo: Workflow para o atendimento de reclamações (modelado com rede de Petri e ACP)



$$\begin{aligned}
 P = & \text{registra} . (\text{contata_cliente} \parallel \text{contata_depto}) . \text{coleta_dados} . \\
 & ((\text{avaliacao_positiva} . \text{reembolsa}) + \\
 & (\text{avaliacao_negativa} . \text{envia_msg})) . \text{arquiva}
 \end{aligned}$$

Modelos formais de workflows

Podem ser analisados sob duas perspectivas: qualitativa e quantitativa

Análise qualitativa

Principais representantes: a *validação* e a *verificação*

- A validação avalia se o sistema se comporta como o esperado
- A verificação avalia se o sistema respeita alguns critérios estruturais
- É possível, por exemplo, verificar se dois modelos de workflows são equivalentes, ou ainda se um dado workflow possui pontos de impasse (*deadlocks*)

Modelos formais de workflows

Podem ser analisados sob duas perspectivas: qualitativa e quantitativa

Análise quantitativa

Principal representante: a *análise de desempenho*

- Feita a partir de modelos estocásticos (como os baseados em cadeias de Markov)
- Prediz tempo de resposta do sistema
- Obtém a probabilidade de execução de uma dada sequência de atividades do workflow
- Identifica gargalos na execução do workflow

Tanto as Álgebras de Processos quanto as RdPs possuem extensões estocásticas, usadas para análises quantitativas.

Modelagem formal de workflows

Por que usá-la?

- Geralmente, a semântica operacional dos conectores e outros construtores presentes nas linguagens de modelagem de workflows não é definida formalmente
 - Isso impossibilita a verificação e a validação dos modelos
 - A partir de modelos formais, é possível fazer análises preditivas qualitativas e quantitativas dos workflows

Classes de formalismos mais usados na modelagem de workflows:

- Redes de Petri (RdPs)
- Álgebras de Processos

Breve introdução às Redes de Petri (RdPs)

- A teoria das Redes de Petri é um dos exemplos mais conhecidos de teoria de ordem parcial para modelagem e análise de sistemas concorrentes
- O conceito foi introduzido por Carl Adam Petri, em sua tese de doutorado em 1962, na Faculdade de Matemática e Física da Universidade de Darmstadt, Alemanha
- São muito utilizadas devido à sua representação gráfica de fácil compreensão e ao seu potencial matemático para a análise de modelos
 - Essas análises incluem verificações de propriedades inerentes aos sistemas concorrentes, como relações de precedência entre eventos, sincronização e existência ou não de impasses

Elementos de uma RdP

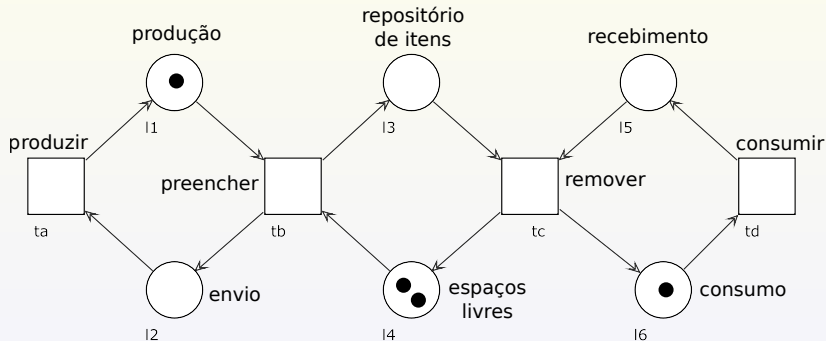
As RdPs são formadas basicamente por dois tipos de componentes

- **Transições** representando ações do sistema
- **Lugares** representando variáveis de estado do sistema

Uma RdP com marcação é formalmente definida pela tupla $RdP = \{L, T, A, P, m_0\}$, em que:

- $L = \{l_1, l_2, \dots, l_L\}$ é um conjunto de lugares;
- $T = \{t_1, t_2, \dots, t_T\}$ é um conjunto de transições;
- $A \subseteq (L \times T) \cup (T \times L)$ é o conjunto de arcos;
- $P : A \rightarrow \mathbb{N}$ é a função peso dos arcos;
- $m_0 = \{m_{01}, m_{02}, \dots, m_{0L}\}$ é a marcação inicial da rede.

Exemplo de RdP: Processos Produtor-Consumidor



Elementos de uma RdP

Os conceitos associados a cada um dos itens de uma RdP são:

- **Lugar** (representado graficamente por um círculo)
 - Modela uma condição que deve ser satisfeita para que o disparo da transição seja realizado
- **Transição** (representada graficamente por um quadrado, retângulo ou barra)
 - Pode ser compreendida como uma ação ou evento
- **Arco orientado**
 - Liga um lugar a uma transição ou vice-versa, encadeando condições e eventos

Elementos de uma RdP

Os conceitos associados a cada um dos itens de uma RdP são:

- **Marca ou Ficha** – representa um recurso disponível
 - O posicionamento dessas fichas nos lugares do grafo constitui a **marcação** da RdP
 - Cada lugar possui 0 ou mais fichas
 - A evolução da marcação permite modelar o comportamento dinâmico do sistema
- **Peso** – cada arco possui um peso associado a ele
 - O peso indica quantas fichas uma transição consome de um lugar de entrada ou quantas fichas uma transição acrescenta em um lugar de saída
 - Quando um arco não possui um peso explicitamente indicado no grafo, considera-se que o seu peso é 1

Marcação de uma RdP

- Uma marcação é uma atribuição de fichas a lugares
- Pode ser representada por um vetor cujo tamanho é o número de lugares na rede: o $i^{\text{ésimo}}$ componente do vetor representa o número de fichas contidas no lugar l_i
- Denotamos por $m(l)$ o número de fichas em um lugar $l \in L$

Evolução da marcação de uma RdP

- A evolução dinâmica da marcação de uma RdP é governada pela ocorrência (disparos) de transições que consomem e produzem fichas
- Regras de **habilitação** e **disparo** estão associadas às transições

Habilitação de uma transição em uma RdP

■ Habilitação

Um transição t está habilitada na marcação m se e somente se:

$$\forall l \in \bullet t, \quad m(l) \geq P(\langle l, t \rangle)$$

onde $\bullet t = \{u \mid \langle u, t \rangle \in A\}$.

Em outras palavras, $\bullet t$ é o conjunto de todos os lugares de entrada de t .

Disparo de uma transição em uma RdP

■ Disparo

O disparo de uma transição t , habilitada na marcação m , produz um marcação m' tal que

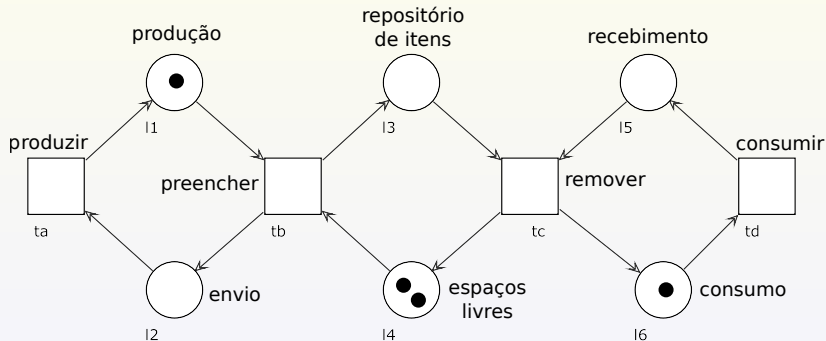
$$m' = m + O(t) - I(t) ,$$

em que

$$I(t) = (P(\langle l_1, t \rangle), P(\langle l_2, t \rangle), \dots, P(\langle l_L, t \rangle)) \quad \text{e} \\ O(t) = (P(\langle t, l_1 \rangle), P(\langle t, l_2 \rangle), \dots, P(\langle t, l_L \rangle))$$

- Essa declaração é normalmente indicada de forma compacta como $m[t]m'$ e podemos dizer que m' é diretamente alcançável a partir de m

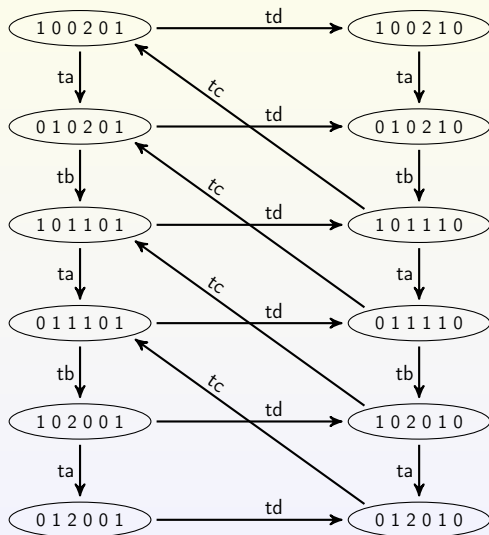
Exemplo de RdP: Processos Produtor-Consumidor



Exemplo de RdP: Processos Produtor-Consumidor

- O produtor só produz um item por vez, ao passo que o consumidor só consome um item por vez
- O produtor só pode enviar um item produzido a um consumidor quando há espaço para armazená-lo, ou seja, quando há ao menos uma ficha em l_4
- O consumidor só pode consumir um item após a sua remoção do repositório (transição t_c). Essa remoção só pode ocorrer quando há itens produzidos e ainda não consumidos, ou seja, quando há ao menos uma ficha em l_3
- Sendo assim, uma ficha que sai de l_4 com o disparo de t_b entra em l_3
- Do mesmo modo, uma ficha que sai de l_3 com o disparo de t_c entra em l_4

Marcações alcançáveis da rede do exemplo



Sobre as marcações de uma RdP

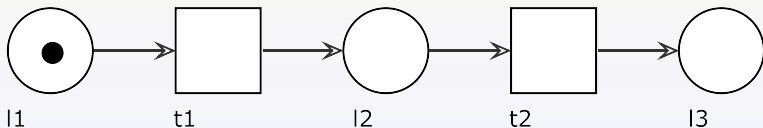
- Cada uma das marcações possíveis da rede representa um possível estado do sistema modelado
- A partir dos estados e da estrutura da rede, é possível verificar propriedades do sistema. Exemplos:
 - Se existe uma transição que não é habilitada por nenhuma marcação da rede, então essa transição é *morta* (ou seja, ela nunca será disparada) $\Rightarrow$ isso pode indicar um erro no projeto do sistema
 - Pode-se querer saber se o sistema pode atingir um determinado estado indesejado. Para isso, identifica-se qual é a marcação da RdP que representa esse estado e caso essa marcação esteja entre as alcançáveis da rede, então estará confirmado que o estado indesejado pode ocorrer no sistema

Mapeamento das estruturas básicas de workflows para RdPs

Sequência ou *Pipeline*

Quando duas atividades devem ser executadas sequencialmente, ou quando uma produz dados que são usados pela outra, então há uma interdependência entre elas

- Em RdPs, isso pode ser modelado conectando as duas atividades (transições) por meio de um lugar



Mapeamento das estruturas básicas de workflows para RdPs

Paralelismo ou distribuição de dados

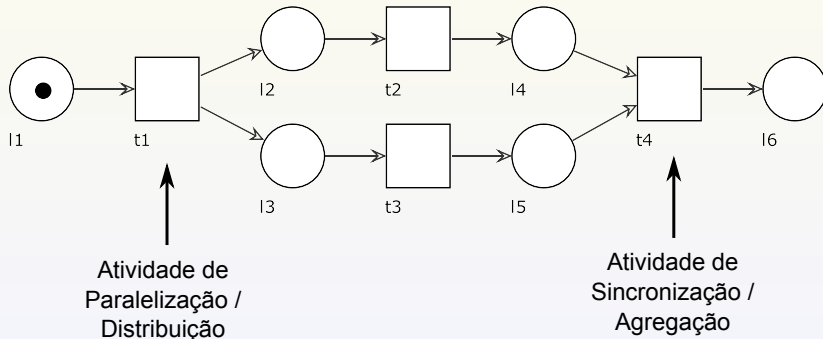
- Modelado por uma atividade (transição) que, quando disparada, inclui uma ficha em cada um dos lugares de entrada das transições que representam as atividades que serão executadas paralelamente ou que processarão os dados distribuídos

Sincronização ou agregação de dados

- É modelada por uma atividade que tem como lugar de entrada os lugares de saída que indicam quando as atividades a serem sincronizadas já foram concluídas ou quando os dados a serem agregados já estão disponíveis para processamento

Mapeamento das estruturas básicas de workflows para RdPs

Paralelismo e sincronização / Distribuição e agregação de dados



Mapeamento das estruturas básicas de workflows para RdPs

Escolha exclusiva

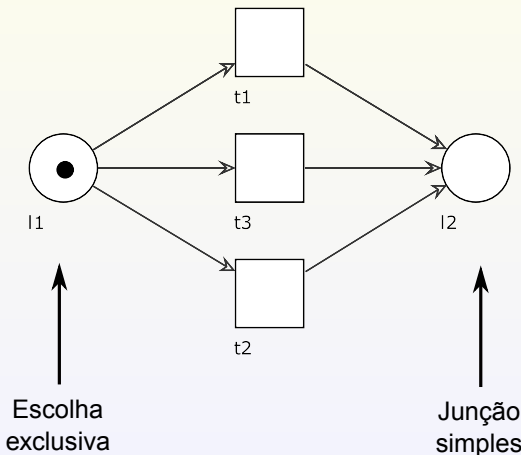
- Pode ser representada como um conjunto de atividades (transições) compartilhando um único lugar de entrada
- Há variações possíveis nessa representação
 - Exemplo: a escolha pode depender de uma regra que pode ser modelada como uma atividade “gerencial”, que quando disparada coloca uma ficha no lugar de entrada da atividade escolhida para execução

Junção simples

- É modelada por um conjunto de atividades (transições) alimentando um mesmo lugar de saída

Mapeamento das estruturas básicas de workflows para RdPs

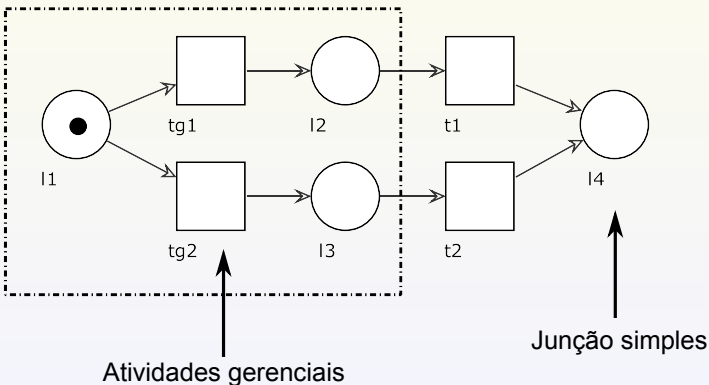
Escolha exclusiva e junção simples



Mapeamento das estruturas básicas de workflows para RdPs

Escolha exclusiva baseada em regras e junção simples

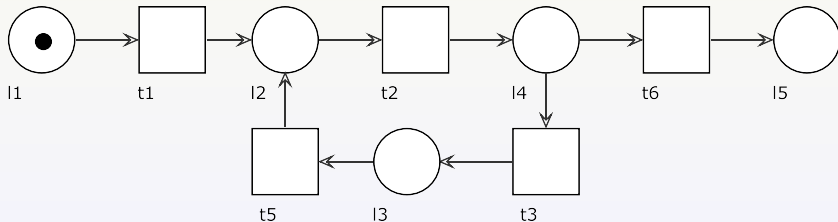
Escolha exclusiva



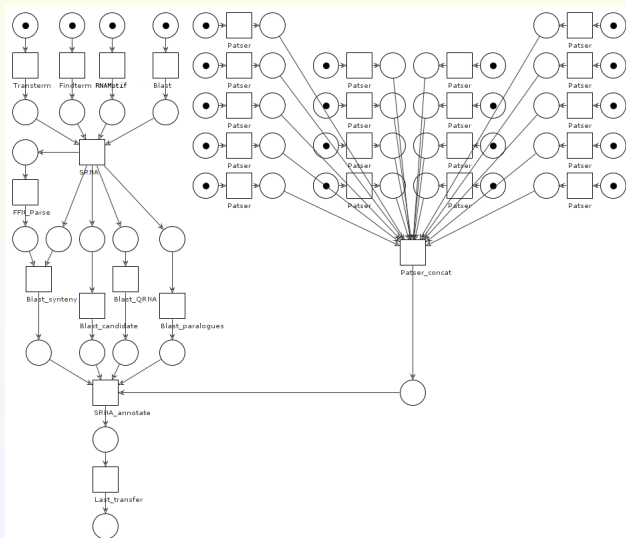
Mapeamento das estruturas básicas de workflows para RdPs

Ciclos

- Geralmente são modelados com o auxílio de escolhas exclusivas e junções simples



O workflow científico SIPHT modelado em RdP



Brevíssima introdução às Álgebras de Processos

- Nas *álgebras de processos* (APs) os processos são representados **algebricamente**, na forma de termos
- Uma AP é composta por um conjunto de **símbolos de ações**, um conjunto de **operadores** e um conjunto de **axiomas** descrevendo as propriedades dos operadores
- O conjunto de axiomas (= leis equacionais) pode também especificar quando dois processos são considerados equivalentes
- Esse arcabouço algébrico possibilita o raciocínio formal sobre **propriedades estruturais e comportamentais** de um modelo de sistema concorrente

APs – Origens

- As bases das álgebras de processos foram desenvolvidas, independentemente, por Milner e Hoare (no anos 70)
- As APs são parcialmente enraizadas nas Redes de Petri, na Teoria de Autômatos e nas linguagens formais
- Milner desenvolveu a álgebra de processos ***Calculus of Communicating Systems*** (CCS), enquanto Hoare definiu a ***Communicating Sequential Processes***(CSP)
- Outras álgebras de processos bem difundidas são a ***Language of Temporal Ordering Specification*** (LOTOS), a ***Algebra of Communicating Processes*** (ACP) e a ***π -Calculus***

Assinatura de uma AP básica

Geralmente consiste em:

- um conjunto finito e não vazio A de **ações atômicas**, representando comportamentos indivisíveis
- um operador binário “+”, chamado de **composição alternativa**

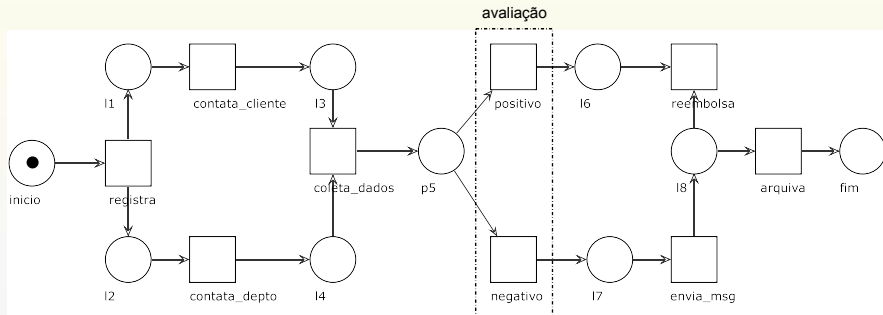
Se os termos fechados t_1 e t_2 representam os processos p_1 e p_2 , respectivamente, então o termo fechado $t_1 + t_2$ representa o processo que executa p_1 ou p_2

- um operador binário “.”, chamado de **composição sequencial**
Se os termos fechados t_1 e t_2 representam os processos p_1 e p_2 , respectivamente, então o termo fechado $t_1.t_2$ representa o processo que executa primeiro p_1 e depois p_2

Outros operadores das APs

- Para modelar cenários que envolvem **paralelismo**, **comunicação** e **iterações**, as álgebras de processos geralmente possuem, além dos operadores básicos, mecanismos para expressar comportamentos mais avançados
- Por exemplo, na *Algebra of Communicating Processes* (ACP) existe:
 - o operador binário “ $\parallel$ ”, chamado de **entrelaçamento**, para expressar paralelismo
 - **equações recursivas**, para representação de ciclos finitos ou infinitos

Exemplo: Workflow para o atendimento de reclamações (modelado com rede de Petri e ACP)



$$\begin{aligned}
 P = & \text{registra} . (\text{contata_cliente} \parallel \text{contata_depto}) . \text{coleta_dados} . \\
 & ((\text{avaliacao_positiva} . \text{reembolsa}) + \\
 & (\text{avaliacao_negativa} . \text{envia_msg})) . \text{arquiva}
 \end{aligned}$$

Modelos formais de workflows

Podem ser analisados sob duas perspectivas: qualitativa e quantitativa

Análise qualitativa

Principais representantes: a *validação* e a *verificação*

- A validação avalia se o sistema se comporta como o esperado
- A verificação avalia se o sistema respeita alguns critérios estruturais
- É possível, por exemplo, verificar se dois modelos de workflows são equivalentes, ou ainda se um dado workflow possui pontos de impasse (*deadlocks*)

Modelos formais de workflows

Podem ser analisados sob duas perspectivas: qualitativa e quantitativa

Análise quantitativa

Principal representante: a *análise de desempenho*

- Feita a partir de modelos estocásticos (como os baseados em cadeias de Markov)
- Prediz tempo de resposta do sistema
- Obtém a probabilidade de execução de uma dada sequência de atividades do workflow
- Identifica gargalos na execução do workflow

Tanto as Álgebras de Processos quanto as RdPs possuem extensões estocásticas, usadas para análises quantitativas.

Escalonamento de Workflows Científicos

Um gostinho do que é Escalonamento

Escalonamento

- é a alocação de recursos (escassos) em função do tempo
- é um processo de decisão que tem como intuito otimizar um ou mais objetivos

Elementos

Recursos máquinas em uma fábrica; pistas em um aeroporto; CPUs em um computador

Tarefas operações de produção; pousos e decolagens; programas de computadores

Objetivos otimizar o tempo do término da última tarefa; minimizar o número de tarefas “atrasadas”

Um exemplo simples: escalonamento em um aeroporto

Uma companhia aérea possui uma frota com diferentes tipos de aeronaves. Dados:

- um conjunto de voos caracterizados por:
 - locais de origem e destino
 - tempo estimado para a decolagem e aterrissagem
 - demanda dos usuários
- uma planilha de custos que indica qual o custo de utilizar um determinado tipo de aeronave em cada uma das rotas

Problema

Determinar como atribuir cada aeronave disponível a uma rota de forma que o lucro da companhia seja maximizado.

Definição (informal)

Definição

Escalonamento se refere a uma alocação, no tempo, de um conjunto limitado de recursos a tarefas. Escalonamento é um processo de decisão que tem como intuito otimizar um ou mais objetivos.⁴

⁴Michael L. Pinedo. *Scheduling. Theory, Algorithms, and Systems*. 4^a ed. Springer, 2012. ISBN: 978-1-4614-1986-0.

Elementos básicos de um problema de escalonamento

- Um conjunto de n tarefas $\mathcal{T} = \{t_1, \dots, t_n\}$
- Um conjunto de m máquinas $\mathcal{M} = \{m_1, \dots, m_m\}$

Encontrar um escalonamento significa encontrar, para cada tarefa, uma alocação de um ou mais intervalos de tempo, em uma ou mais máquinas. O problema de escalonamento correspondente é encontrar um escalonamento que satisfaça um determinado conjunto de restrições.

Exemplo de um escalonamento

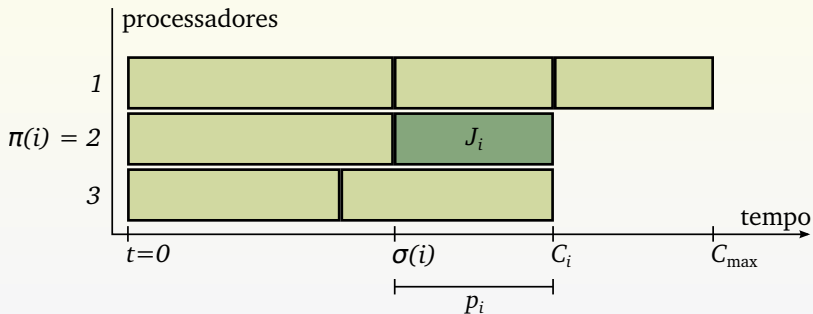
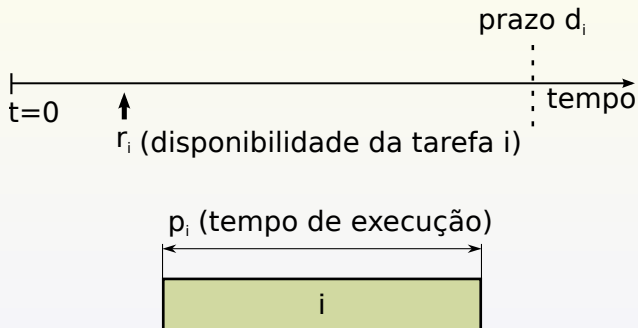


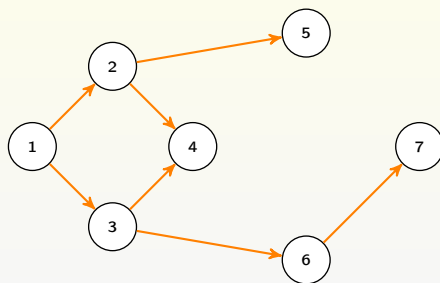
Figura : Exemplo de escalonamento representado por um diagrama de Gantt

Notações



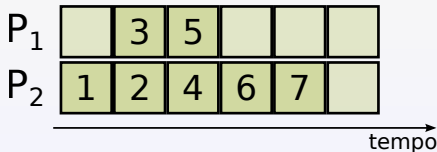
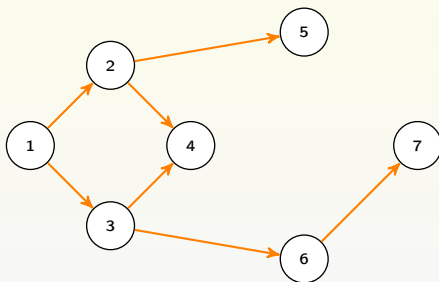
O DAG de dependências

De um workflow a um escalonamento



O DAG de dependências

De um workflow a um escalonamento



Informações sobre as tarefas

Associadas às tarefas, temos as seguintes informações:

tempo de execução (p_{ij}) tempo necessário para a execução da tarefa t_i na máquina m_j . Denotado apenas por p_i se todas as máquinas forem idênticas.

data de disponibilidade (r_i) instante em que a tarefa t_i se torna disponível para ser executada

prazo (d_i) instante de tempo no qual a tarefa t_i deve estar pronta

peso (w_i) normalmente indica um fator de prioridade da tarefa t_i

Informações sobre as tarefas

Associadas às tarefas, temos as seguintes informações:

tempo de execução (p_{ij}) tempo necessário para a execução da tarefa t_i na máquina m_j . Denotado apenas por p_i se todas as máquinas forem idênticas.

data de disponibilidade (r_i) instante em que a tarefa t_i se torna disponível para ser executada

prazo (d_i) instante de tempo no qual a tarefa t_i deve estar pronta

peso (w_i) normalmente indica um fator de prioridade da tarefa t_i

Informações sobre as tarefas

Associadas às tarefas, temos as seguintes informações:

tempo de execução (p_{ij}) tempo necessário para a execução da tarefa t_i na máquina m_j . Denotado apenas por p_i se todas as máquinas forem idênticas.

data de disponibilidade (r_i) instante em que a tarefa t_i se torna disponível para ser executada

prazo (d_i) instante de tempo no qual a tarefa t_i deve estar pronta

peso (w_i) normalmente indica um fator de prioridade da tarefa t_i

Informações sobre as tarefas

Associadas às tarefas, temos as seguintes informações:

tempo de execução (p_{ij}) tempo necessário para a execução da tarefa t_i na máquina m_j . Denotado apenas por p_i se todas as máquinas forem idênticas.

data de disponibilidade (r_i) instante em que a tarefa t_i se torna disponível para ser executada

prazo (d_i) instante de tempo no qual a tarefa t_i deve estar pronta

peso (w_i) normalmente indica um fator de prioridade da tarefa t_i

Classificação $\alpha \mid \beta \mid \gamma$

Problemas de escalonamento podem ser classificados utilizando a notação:

$$\alpha \mid \beta \mid \gamma$$

Onde:

α descreve os recursos disponíveis

β descreve as tarefas a serem executadas

γ descreve o critério de otimização

Recursos (α) – Máquinas paralelas

1 uma única máquina

P ou Pm m máquinas paralelas idênticas

- cada tarefa t_i pode ser processada em qualquer uma das m máquinas por p_i unidades de tempo
- se uma tarefa só puder ser executada em um subconjunto das máquinas, a restrição será indicada no campo β

$P\infty$ ou $\overline{P}$ número ilimitado de máquinas idênticas

Recursos (α) – Máquinas paralelas

Qm máquinas paralelas uniformes

- m máquinas que operam com diferentes velocidades (a máquina m_j opera com velocidade v_j)
- $p_{ij} = p_i/v_j$

Rm máquinas paralelas não-relacionadas

- m máquinas diferentes em paralelo
- $p_{ij} = p_i/v_{ij}$, onde v_{ij} é a velocidade da tarefa t_i na máquina m_j

Note que Pm é um caso particular de Qm , que por sua vez é um caso particular de Rm .

Tarefas (β)

r_i data de disponibilidade

- se r_i não estiver no campo β , tarefas podem ser iniciadas a qualquer momento
- se r_i estiver no campo β , então a tarefa t_i não pode ser iniciada antes do tempo r_i

$pmtn$ interrupção

o processamento de uma tarefa pode ser interrompido e retomado do ponto onde parou, em qualquer máquina

s_{ik} tempo de preparação da máquina

uma máquina que executou uma tarefa t_i precisa de s_{ik} unidades de tempo para ser preparada antes de executar uma outra tarefa t_k

Tarefas (β)

prec relações de precedência

uma relação de precedência entre duas tarefas indica que uma tarefa não pode ser iniciada antes do término da execução da outra. Representadas por um grafo de precedências, onde um vértice representa uma tarefa e um arco entre t_i e t_k indica que t_k só pode ser executada após o termino de t_i .

c_{ik} tempo de comunicação entre máquinas

a tarefa t_k depende do resultado da execução de t_i . Se t_k não for escalonada na mesma máquina que t_i , então deverá esperar mais c_{ik} unidades de tempo antes de poder ser executada.

Critério de otimização (γ)

Notação

C_i tempo de término da tarefa t_i

$f_i(C_i)$ custo de uma tarefa t_i que termina sua execução no instante C_i

Custo de um escalonamento

Na maior parte dos casos, corresponde ao máximo:

$$\max\{f_i(C_i) \mid 1 \leq i \leq n\}$$

ou à soma de custos:

$$\sum_{i=1}^n f_i(C_i)$$

Critério de otimização (γ)

Funções de custo mais comuns:

$C_{\max}$ *makespan*

Definido por $C_{\max} = \max\{C_1, \dots, C_n\}$, é o instante de tempo do término da última tarefa

$\sum C_i$ **tempo de total de término**

é definido por $\sum C_i = \sum_{i=1}^n C_i$

$\sum w_i C_i$ **tempo de total de término ponderado**

é definido como a soma dos instantes de término com seus pesos: $\sum w_i C_i = \sum_{i=1}^n w_i C_i$

Critério de otimização (γ)

Outras funções comuns para o custo de tarefas:

$L_i = C_i - d_i$ *lateness* (grau de atraso, pode ser negativo)

$E_i = \max\{0, d_i - C_i\}$ *earliness* (grau de “adiantamento”)

$T_i = \max\{0, C_i - d_i\}$ *tardiness* (grau de atraso efetivo)

$U_i = 0$ se $C_i \leq d_i$; 1 c.c. penalidade unitária por atraso

Critério de otimização (γ)

Outros critérios de otimização comuns:

$\sum w_i T_i$ *tardiness* total ponderada

$\sum w_i U_i$ total ponderado de tarefas atrasadas

$\max L_i$ atraso máximo

Da mesma forma que denotamos $C_{\max} = \max C_i$, também é comum denotar os valores máximos dos outros critérios como: $L_{\max}$, $E_{\max}$, $T_{\max}$ e $U_{\max}$.

Na prática

Cientistas tem a disposição três tipos de plataformas:

- Aglomerados (*clusters*) de computadores
- Grades de computação
- Plataformas de computação em nuvem

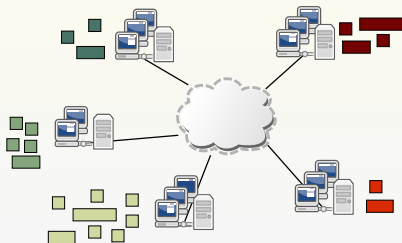
Aglomerados de computadores

Um conjunto de computadores (independentes) que trabalham juntos como se fossem uma só. Grosso modo⁵, com as seguintes características:

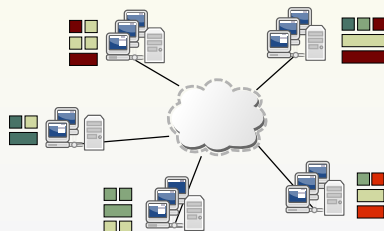
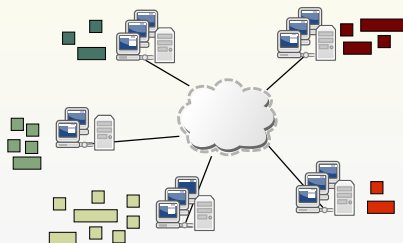
- Hardware parecido (em geral, consideramos os recursos uniformes $\alpha = P_m$)
- Fisicamente próximas
- Interligadas por uma rede rápida de interconexão (ex: InfiniBand)
- Beowulf, máquinas em um mesmo laboratório trabalhando juntas, etc.

⁵Não existe uma definição precisa para o que é um aglomerado.

Grades de computação



Grades de computação

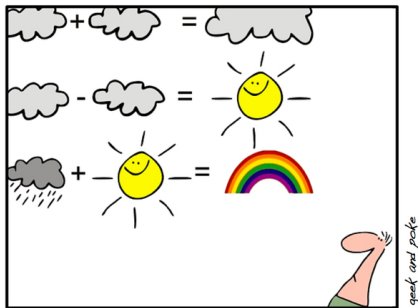


Grades de computação

Conjunto de computadores e aglomerados que podem ser utilizados por qualquer usuário. A ideia é que qualquer um possa utilizar os recursos computacionais ociosos dos outros. Tipicamente:

- Os recursos estão dispersos geograficamente
- Os recursos são heterogêneos
- Computação disponível como um serviço, sem a preocupação de onde o serviço é oferecido. Analogia com a rede elétrica (de qual usina hidrelétrica vem a eletricidade desta tomada?)

Plataformas de computação em nuvem



SIMPLY EXPLAINED - PART 17:
CLOUD COMPUTING

Figura : O que é computação em nuvem?

Plataformas de computação em nuvem

*É um modelo que possibilita **acesso ubíquo**, de forma conveniente e **sob demanda** à um conjunto de recursos de computação configuráveis (por exemplo, redes, servidores, dispositivos de armazenamento, aplicações e outros serviços) **que podem ser rapidamente provisionados e dispensados** com o mínimo esforço de gestão ou interação do prestador de serviço.*

Instituto Nacional de Padrões e Tecnologia dos EUA (NIST):
<http://csrc.nist.gov/publications/PubsSPs.html#800-145>

Plataformas de computação em nuvem

Características:

- Recursos *quasi* ilimitados
- Podem ser alugados ou devolvidos a qualquer momento
- Difícil prever o desempenho dos recursos
- Investimento em capital vs. custos operacionais (CAPEX vs. OPEX)

Plataformas de Computação em Nuvem

Classificação quanto à estrutura

- *Privadas*: recursos computacionais próprios;
- *Comunitárias*: recursos de várias organizações;
- *Públicas*: recursos de fornecedores comerciais (ex.: *Amazon Elastic Compute Cloud* ⁶ e *Microsoft Azure Services* ⁷;
- *Híbridas*: combinações das estruturas anteriores.

⁶Amazon AWS: <http://aws.amazon.com>

⁷Microsoft Azure: <http://azure.microsoft.com>

Balanceamento de carga

Em geral, estamos interessados em balancear a carga entre os recursos disponíveis (processadores, arquivos, etc.)

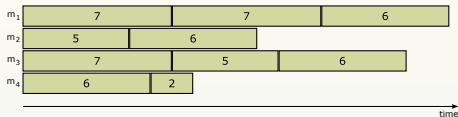


Figura : desbalanceado

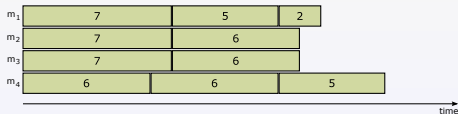


Figura : balanceado

O problema do Balanceamento de Carga

Problema: dado um conjunto de n tarefas caracterizadas pela sua duração p_i para $1 \leq i \leq n$, o objetivo é minimizar o tempo máximo de término (carga) em todas as máquinas.

O objetivo é chamado *makespan* e é denotado por C_{max} .

Complexidade do problema de balanceamento de carga

O problema do balanceamento de carga é NP-difícil

Ele contém o problema *Partição* como um subproblema. Por isso, em geral procuramos por *soluções aproximadas*.

FPTAS

Existe um (*Fully Polynomial Time Approximation Scheme* – i.e. uma $(1 + \epsilon)$ -aproximação com ϵ tão pequeno quanto se queira). Mas para escalonar workflows, procuramos por algoritmos que sejam **rápidos**.

Algoritmos de lista

List scheduling (LS)

Princípio

Construir uma lista com as tarefas prontas para serem executadas e alocá-las tão logo um recurso fique disponível.

Análise dos algoritmos de lista

Proposição

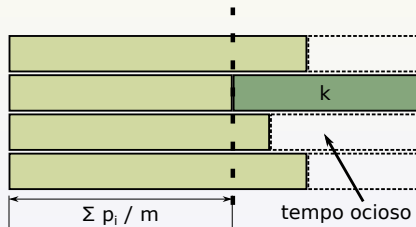
Todo algoritmo de lista é uma 2-aproximação para o problema do balanceamento de carga.

Mais precisamente, é possível provar que:

$$C_{\max} \leq (2 - \frac{1}{m})C_{\max}^*$$

Ideia da prova do fator 2

Seja $p_{\max} = \max(p_i)$. A ideia da prova é considerar a tarefa que termina por último no escalonamento. Seja k seu índice.



Refinamento: a regra LPT

Longest Processing Time first

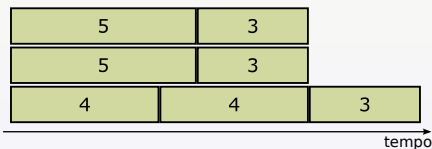
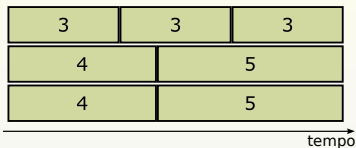
Ideia

Ordenar a lista de tarefas prontas em ordem não-crescente de tempo de execução.

Proposição

O algoritmo LPT é uma $\frac{4}{3} - \frac{1}{3m}$ -aproximação para o problema de balanceamento de carga.

Exemplo de uso do LPT



O exemplo mostra que o fator de aproximação de $\frac{4}{3} - \frac{1}{3m}$ é justo.

SPT

Shortest Processing Time first

Nem sempre estamos interessados em minimizar o makespan.

Por exemplo: e se os resultados intermediários pudessem ser utilizados imediatamente?

Nesse caso, queremos minimizar o tempo médio dos termos das tarefas.

O algoritmo SPT — que escalona as tarefas em ordem não-decrescente de tempo de execução (o contrário do LPT) — **é ótimo.**

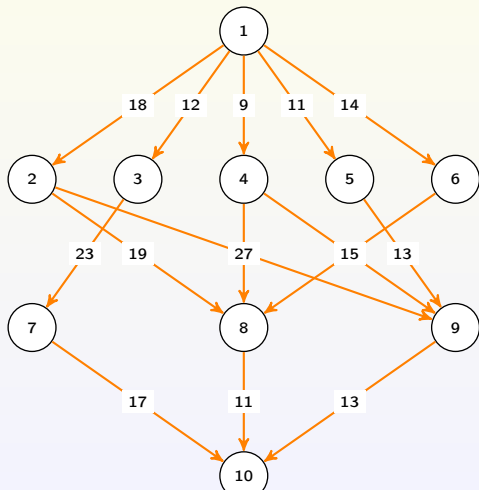
HEFT

Heterogeneous Earliest Finish Time

- Algoritmo para escalonamento em recursos **heterogêneos**
- Funciona em duas etapas:
 - Priorização das tarefas** : as tarefas são ordenadas em ordem não-crescente de custo (tempo de execução e comunicação)
 - Seleção dos recursos** : cada tarefa é alocada, de acordo com a ordem de prioridade, na máquina ociosa que a terminar primeiro.

HEFT

Exemplo

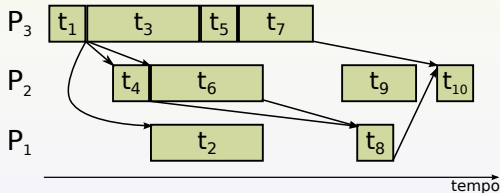


Tarefa	P_1	P_2	P_3
1	14	16	9
2	13	19	18
3	11	13	19
4	13	8	17
5	12	13	10
6	13	16	9
7	7	15	11
8	5	11	14
9	18	12	20
10	21	7	16

Tabela : Custos
computacionais

HEFT

Escalonamento resultante



Tarefa	P_1	P_2	P_3
1	14	16	9
2	13	19	18
3	11	13	19
4	13	8	17
5	12	13	10
6	13	16	9
7	7	15	11
8	5	11	14
9	18	12	20
10	21	7	16

Tabela : Custos
computacionais

Algoritmos com um único objetivo

- Os algoritmos clássicos foram desenvolvidos para plataformas computacionais bem conhecidas
 - aglomerados (*clusters*) de computadores
 - grades de computação
 - etc.
- Objetivo bem definido: *makespan*, tempo médio de término de tarefas, etc.
- Ambiente estático

Novas plataformas são mais dinâmicas e cooperativas

- Grades cooperativas
 - Diversas organizações compartilham seu recursos e competem pela plataforma
 - O que privilegiar: o desempenho de cada organização ou o desempenho global do sistema?
- Computação em nuvem
 - Uma *grande* quantidade de recursos computacionais é oferecida comercialmente
 - O que privilegiar: o desempenho da execução ou o custo do aluguel dos recursos?

Como comparar soluções diferentes?

Não há uma única solução que seja ótima!

Soluções com bom compromisso

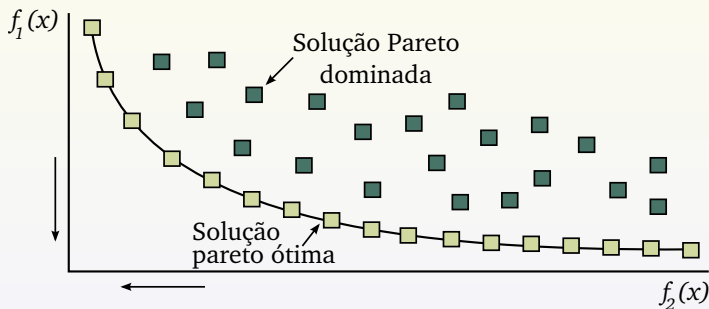


Figura : Conjunto de soluções de Pareto para um problema biobjetivo

Sistemas de Gerenciamento de Workflows Científicos

Sistema de gerenciamento de workflows científicos (SGWC)

Definição

É um sistema que permite definir, modificar, gerenciar, executar e monitorar workflows científicos, utilizando um nível de abstração que oculta dos cientistas as technicalidades envolvidas nessas tarefas.

Funcionalidades oferecidas pelos SGWCs

- Interface de usuário personalizável e interativa
- Ferramentas de apoio à reprodutibilidade do experimento
- Integração a ferramentas e serviços distribuídos heterogêneos
- Apoio à execução em sistemas computacionais de alto desempenho
- Monitoramento da execução de um workflow e tratamento de falhas na execução

Subsistemas que compõem os SGWCs

- Subsistema de projeto
- Subsistema de visualização e apresentação
- Subsistema de execução
- Subsistema de gerenciamento de atividades
- Subsistema de gerenciamento de dados
- Subsistema de gerenciamento de proveniência
- Subsistema de monitoramento

Subsistema de projeto de workflows

- Permite a criação e a modificação de especificações (modelos)
- Produz especificações em formato digital, descritas em uma linguagem de modelagem de workflows
- Em alguns casos, estabelece a interoperabilidade entre diferentes SGWCs
 - pode usar linguagens de representação padronizadas

Subsistema de apresentação e representação

- Permite visualizar o workflow e os dados produzidos por ele em diferentes formatos
- Desejável:
 - uma apresentação que facilite a análise dos dados e de sua proveniência
 - possibilite a extração de conhecimento a partir do grande volume de dados e metadados gerados pela execução de diversos experimentos

Subsistema de execução de workflows

- Gerencia a execução dos workflows
- Cria e executa instâncias de workflows
- Ampara-se no modelo digital dos workflows
- Escalona as atividades nos recursos computacionais disponíveis
- Controla a execução concorrente de múltiplas instâncias
- Faz uso do *subsistema de gerenciamento de atividades* para lidar a possível heterogeneidade do sistema distribuído
- Faz uso do *subsistema de gerenciamento de dados* para gerenciar os dados usados e produzidos pelas atividades e lidar com problemas como o da heterogeneidade e grande volume dos dados

Subsistema de gerenciamento de proveniência

Subsistema de monitoramento

Responsáveis por:

- Representação, armazenamento e consulta dos dados e metadados de proveniência
- Monitoramento do status da execução das instâncias, detectando falhas e auxiliando no seu tratamento

Alguns SGWCs “famosos”

ASKALON <http://www.askalon.org/>

Kepler <http://kepler-project.org/>

Pegasus <http://pegasus.isi.edu/>

Taverna <http://www.taverna.org.uk/>

Triana <http://www.trianacode.org/>

SciCumulus <http://sourceforge.net/projects/scicumulus/>

E alguns sistemas “famosos” que não são SGWCs “clássicos”

Galaxy <http://galaxyproject.org/>

Swift <http://swift-lang.org/>

VisTrails <http://www.vistrails.org/>

Taverna

- Parte do projeto myGrid [<http://www.mygrid.org.uk/>]
- Motivado por problemas de Ciências Biológicas, mas atualmente usado em várias outras áreas (ciências sociais, música, astronomia, etc.)
- Permite a modelagem de workflows a partir de *componentes*, i.e., serviços **remotos** (serviços web) ou **locais** (*scripts*) que podem ser reutilizados

Taverna

Modelo de workflow

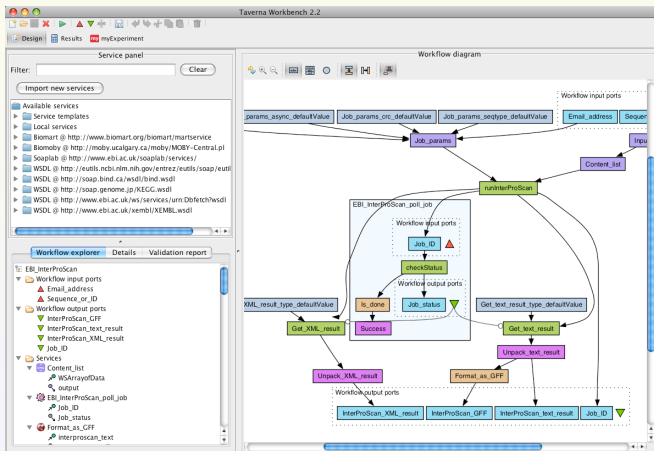
- Um workflow Taverna é especificado como um grafo direcionado
- Os vértices do grafo (*processadores*) são os serviços que implementam as atividades
- Cada arco conecta um par entrada-saída e denota dependência de dados entre as atividades
- Há também *ligações de controle* que indica que uma atividade só pode terminar depois da execução de outra

Taverna

Escalonamento

- Os serviços executados pelos processadores devem ser implantados manualmente antes da execução do workflow
- Não há necessidade de mapear a execução das atividades aos recursos computacionais
- O Taverna habilita uma atividade assim que ela tiver algum dado para processar. Cada atividade é executada em uma *thread* separada

Taverna Workbench



Taverna

Scufl

- Sculfl: Simplified Conceptual Workflow Language
- Linguagem textual baseada em XML
- Especializada na representação de fluxo de dados
 - a construção de *pipelines* é a atividade básica
 - distribuição, agregação e redistribuição são modeladas como múltiplas entradas e saídas ligadas às atividades
- Além das ligações de controle, Scufl possui apenas operações de paralelismo e sincronização.
 - escolhas, junções simples e iterações só podem ser expressas por meio do fluxo de dados
- A linguagem não possui condicionais, mas o Taverna disponibiliza atividades FailIfFalse e FailIfTrue

myExperiment

- O Taverna está integrado ao portal myExperiment [<http://www.myexperiment.org/>]
- O portal permite que cientistas do mundo inteiro compartilhem seus workflows com outros pesquisadores
- Basta a URL do workflow; o Taverna se encarrega de baixá-lo e executá-lo automaticamente
- Em julho de 2014, o portal disponibilizava cerca de 2500 workflows

Exemplos de workflows do Taverna

Clique no ícone abaixo para obter os arquivos utilizados na demonstração do Taverna



Kepler

- SGWC de código-aberto desenvolvido por pesquisadores da Universidade da Califórnia (Davis, Santa Barbara e San Diego)
- Uma metáfora de **diretor** / **ator** para representar visualmente os componentes de um workflow
 - os atores especificam *que* processamento é feito
 - o diretor especifica *quando* o processamento ocorre

Kepler

Diretor

- Cada modelo de computação é representado por um tipo de diretor
 - o SDF Director executa o workflow de forma síncrona, um componente por vez
 - o PN Director executa todos os componentes em paralelo
 - etc.
- O projetista do workflow é quem decide qual(is) diretor(es) será(ão) usado(s)

Kepler

Atores

- Especifica todos os tipos de ação que o workflow pode realizar
- Contêm *portas* que definem consumo ou produção de dados, interligadas por *canais*
- Algumas ações disponibilizadas por atores do Kepler:
 - acesso a serviços web (WebService),
 - leitura de dados do disco (FileFetcher, FileStager, etc.),
 - acesso a serviços de notificação (Email)
 - composição de vários atores (Composite)
 - etc.



Askalon

- Desenvolvido pelo grupo de sistemas distribuídos da Uni. de Innsbruck, Áustria
- Utiliza a linguagem de modelagem AGWL (*Abstract Grid Workflow Language*)
- Permite a criação de modelos de forma textual (diretamente em AGWL) ou gráfica

AGWL

- Linguagem textual baseada em XML
- Possui construções tanto de fluxo de controle, como de fluxo de dados
- É uma linguagem tipificada: permite definir os tipos de entradas e saídas (integer, float, boolean, string, file e collection)

AGWL

Fluxo de controle

AGWL possui construções que permitem modelar o fluxo de controle. Entre elas:

sequence elenca atividades que devem ser executadas em sequência

parallel indica atividades que podem ser executadas simultaneamente

switch modela escolhas exclusivas e junções simples

while, for e forEach iterações nas atividades do workflow ou em conjuntos de dados

parallelFor e parallelForEach cria iterações que podem ser executadas em paralelo

AGWL

Fluxo de dados

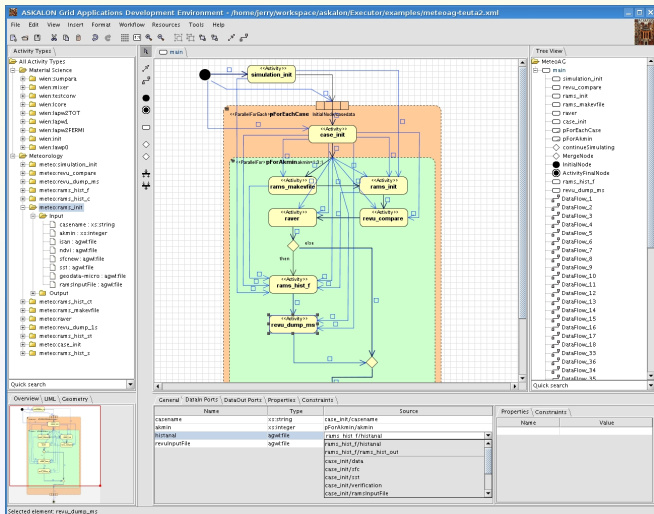
- Permite correlacionar as entradas e saídas de dados das atividades
- Uma atividade pode ter múltiplas entradas e múltiplas saídas

Dessa forma:

- um *pipeline* é criado com um modelo onde a saída de uma atividade é a entrada de outra.
- uma *agregação* é definida por uma atividade que possui múltiplas entradas de dados
- uma *redistribuição* é definida por uma atividade com múltiplas entradas e múltiplas saídas de dados

Askalon

Exemplo de workflow na interface gráfica



Exemplo de AGWL

```

<cgwd name="Wien2K">
  <cgwdInput>
    <dataIn name="startInput" type="agwl:file"
      source="gsiftp://.../.../WIEN2K/atype/STARTINPUT.txt"/> ...
  </cgwdInput>
  <cgwdBody>
    <while name="whileConv">
      <dataLoops>
        <dataLoop name="overflag" type="xs:boolean"
          initSource="Wien2K/overflag" loopSource="MIXER/overflag"/>
        </dataLoops>
        <loopBody>
          <activity name="LAPW0" type="wien:LAPW0">
            <dataIns>
              <dataIn name="startInput" type="..." source="Wien2K/startInput"/> ...
            </dataIns>
            <dataOuts>
              <dataOut name="kpoints" type="xs:integer"/> ...
            </dataOuts>
          </activity>
          <parallelFor name="pforLAPW1">
            <loopCounter name="lapw1Index" type="..." from="1" to="LAPW0/kpoints"/>
            <loopBody>
              <activity name="LAPW1" type="wien:LAPW1" .../>
            </loopBody>
            <dataOuts .../>
          </parallelFor>
        </loopBody>
      </while>
    </cgwdBody>
  </cgwd>

```

Principais serviços do sistema

Escalonador define o mapeamento das atividades aos recursos

Motor de execução gerencia a execução e provê tolerância a falhas

Gerenciador de recursos controla os componentes de software e hardware disponíveis

Predição de desempenho mantém dados de execuções passadas para estimar o tempo de execução e transmissão de dados

Algoritmos de escalonamento

- Algoritmo padrão: HEFT
 - Os custos são estimados com o módulo de predição de desempenho
- *Myopic* algoritmo de lista que prioriza as tarefas pela ordem de chegada
- *Min-Min* e *Max-Min* algoritmos similares ao SPT e LPT, respectivamente, para recursos heterogêneos

Heurística não garantida

O Askalon oferece também um *algoritmo genético*, que permite a otimização de problemas multiobjetivo (mas sem garantia de desempenho)

Pegasus

- O Pegasus (*Planning for Execution in Grids*⁸) é um SGWC concebido para execução de workflows em plataformas distribuídas
- A partir da descrição do modelo do workflow e da informação dos recursos, o Pegasus faz o escalonamento, cria todas as tarefas necessárias para a implantação e execução do workflow
- O mesmo workflow pode ser executado em aglomerados, grades e plataformas de computação em nuvem sem modificações

⁸Ewa Deelman et al. "Pegasus: A framework for mapping complex scientific workflows onto distributed systems". Em: *Scientific Programming* 13.3 (2005), pp. 219–237.

Pegasus

Principais características

Portabilidade um mesmo workflow pode ser executado em diferentes plataformas

Desempenho o escalonador pode reordenar, agrupar e priorizar tarefas para melhorar o desempenho da execução

Escalabilidade o Pegasus executa workflows com até um milhão de tarefas

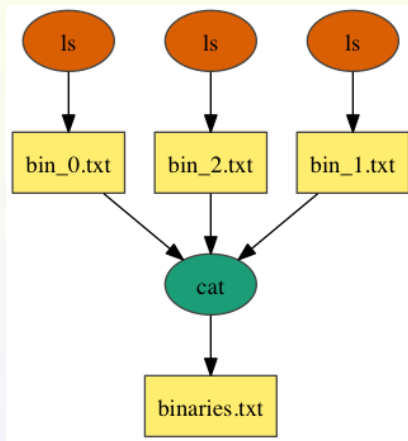
Gerenciamento dos dados o Pegasus é responsável por replicar, transferir e recuperar informações geradas pelas tarefas

Confiabilidade o Pegasus monitora a execução das tarefas e as reexecuta em caso de erros

DAX

- A descrição do workflow é fornecida como um DAG descrito em XML, na linguagem DAX (*Directed Acyclic Graph in XML*)
- O DAX representa o fluxo de processamento e dependências entre as tarefas
- Cada atividade possui um identificador e metadados sobre seus arquivos de entrada e saída
- A linguagem DAX é puramente orientada a fluxo de dados

Exemplo de workflow



Exemplo do workflow em DAX

```

<adag>
  <job id="ID0000001" name="ls">
    <argument>l /bin</argument>
    <stdout name="bin_0.txt" link="output"/>
    <uses name="bin_0.txt" link="output"/>
  </job>
  <job id="ID0000002" name="ls">
    <argument>l /usr/bin</argument>
    <stdout name="bin_1.txt" link="output"/>
    <uses name="bin_1.txt" link="output"/>
  </job>
  <job id="ID0000003" name="ls">
    <argument>l /usr/local/bin</argument>
    <stdout name="bin_2.txt" link="output"/>
    <uses name="bin_2.txt" link="output"/>
  </job>
  <job id="ID0000004" name="cat">
    <argument>
      <file name="bin_0.txt"/>
      <file name="bin_1.txt"/>
      <file name="bin_2.txt"/>
    </argument>
    <stdout name="binaries.txt" link="output"/>
    <uses name="bin_2.txt" link="input"/>
    <uses name="bin_0.txt" link="input"/>
    <uses name="binaries.txt" link="output" register="false" transfer="true"/>
    <uses name="bin_1.txt" link="input"/>
  </job>
</adag>

```

Pegasus

Subsistemas do Pegasus

O sistema é formado por 3 componentes principais:

- *Mapeador (PegasusMapper)* é o escalonador do Pegasus, responsável por encontrar o software, dados e recursos computacionais necessários para a execução do workflow
- *Motor de Execução (DAGMan)* monitora o estado da execução do workflow e controla quais atividades estão prontas para serem executadas
- *Gerenciador de tarefas (Condor-G)* recebe as requisições do DAGMan e gerencia as tarefas do workflow, supervisionando sua execução nos recursos locais e remotos

Pegasus

Escalonamento

- Realizada pelo PegasusMapper
- Capaz de realizar otimizações baseadas no modelo e estado de execução do workflow
 - ex: se o dado gerado por uma atividade já estiver disponível, a atividade não é executada
- Pode agrupar atividades em um mesmo recurso para evitar movimentação de dados

Pegasus

Algoritmos de escalonamento

Random os recursos são escolhidos ao acaso. É a heurística padrão.

RoundRobin os recursos são selecionados sequencialmente, seguindo uma dada ordem

Group grupos de atividades definidas no modelo são mapeadas no mesmo recurso (escolhido ao acaso)

HEFT o bom e velho HEFT para recursos heterogêneos (slide 140)

Desafios Atuais para a Pesquisa na Área de Workflows Científicos

Limitações dos SGWCs atuais (1)

Inabilidade de lidar com fluxos de grandes volumes de dados

- A maioria dos SGWCs só é capaz de gerenciar conjuntos de dados que podem ser armazenados como arquivos comuns, que cabem em uma única máquina
 - Arquivos assim podem ser facilmente transmitidos de um nó para outro quando necessário.

Problema: as transferências de dados entre atividades podem dominar o tempo de execução dos workflows.

Limitações dos SGWCs atuais (2)

Não se beneficiam suficientemente do “poder” das Nuvens

- Os SGWCs vêm usando as nuvens como grades
 - Os recursos são alocados antes do início da execução do workflow e não mudam mais durante toda a execução

Problema: não exploram a elasticidade e escalabilidade de recursos nas nuvens.

Superando as limitações dos SGWCs atuais

Para lidar com as limitações, é preciso soluções avançadas de gerenciamento de recursos.

- Escalonamentos multiobjetivos

- No caso da limitação 1: o objetivo é minimizar o tempo de execução e a quantidade de dados
- No caso da limitação 2: o objetivo é minimizar o tempo de execução e o custo total de alocação dos recursos

Problema: não existe solução ótima para esse tipo de otimização

- Mas existe um conjunto de soluções que oferecem algum tipo de compromisso entre os objetivos

Tópicos de pesquisa atuais

Sobre escalonamento de workflows

Proposição de heurísticas que:

- agregam os múltiplos objetivos em um único objetivo (nem sempre isso é possível)
- tentam otimizar um dos objetivos impondo algum limite máximo para os outros objetivos

Objetivo: encontrar soluções próximas às soluções ótimas de Pareto em tempo hábil, para que o escalonamento não se torne um gargalo no sistema.

Tópicos de pesquisa atuais

O uso de plataformas colaborativas (como as nuvens)

- Acesso simultâneo por múltiplos usuários;
- Imprevisibilidade do desempenho, causada pelo compartilhamento de recursos físicos entre diferentes máquinas virtuais.

O que precisa ser feito

- Criação de algoritmos adaptativos para a execução de workflows
 - Ideia: ajustar dinamicamente o número de recursos em utilização para a execução dos workflows

Alguns grupos que pesquisam sobre Workflows Científicos

No Brasil

- Universidade Federal do Rio de Janeiro (PESC-COPPE-UFRJ), em parceria com o Laboratório Nacional de Computação Científica (LNCC)
 - Criadores das ferramentas Chiron e SciCumulus
- Universidade Estadual de Campinas (IC-UNICAMP)
- Universidade de São Paulo (DCC-IME-USP)

Alguns grupos que pesquisam sobre Workflows Científicos

Fora do Brasil

- University of Southern California (Information Sciences Institute)
 - Profa. Ewa Deelman
 - Criadores do Pegasus
- University of Manchester (School of Computer Science)
 - Profa. Carole Goble
 - Criadores do Taverna, myGrid, myExperiment
- University of Chicago (Computation Institute)
 - Prof. Ian Foster
 - Criadores do conceito de grade e do projeto Globus
- University of Innsbruck (Institute of Computer Science)
 - Prof. Thomas Fahringer
 - Criadores do ASKALON

Obrigado!

Kelly R. Braghetto

`www.ime.usp.br/~kellyrb`

Daniel Cordeiro

`www.ime.usp.br/~danielc`