Rumo à Integração da Álgebra de Workflows com o Processamento de Consulta Relacional

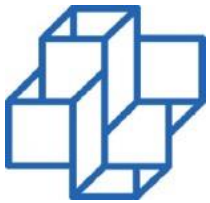
João Ferreira[1], Jorge Soares[1], Fabio Porto[2], Esther Pacitti[3], Rafaelli Coutinho[1], Eduardo Ogasawara[1]

[1] CEFET/RJ – Centro Federal de Educação Tecnológica Celso Suckow da Fonseca

[2] LNCC – Laboratório Nacional de Computação Científica

[3] Inria & University of Montpellier

CEFET/RJ

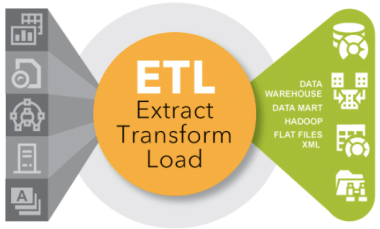


LNCC

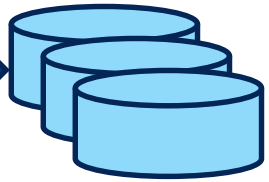
Cenário de Análise de Dados: Data Science Workflows

Este cenário pode ser apoiado por linguagens de programação centrada em dados

1. Dados coletados



2. Grande volume de dados produzidos ...



3. Análise exploratória

4. ...que precisam ser processados por rotinas de MD em ambientes Hadoop/Spark

5. Resultados são analisados

Os cientistas de dados precisam executar tarefas de MD usando diferentes valores de parâmetros

Ferreira etl al (2018), "Rumo à Otimização de Operadores sobre UDF no Spark", BreSci 2018.

DISC - Data Intensive Scalable Computing

- Convergência de aplicações e infraestrutura
  - Vários cenários de aplicações nas Ciências-Empresas-Governos
  - Amplo espectro de ferramentas para análise de dados
- Há demandas em aberto
  - Otimização de execução, Rastreabilidade, *Steering*
- Sistemas de workflows têm primitivas para execução paralela
  - Programação ad hoc e trabalhosa
  - Dependente da especificação feita pelo usuário
  - Apresentam UDFs

Bryant, R. E., (2011), "Data-intensive scalable computing for scientific applications", *Computing in Science and Engineering*, v. 13, n. 6, p. 25–33.
Valduriez et al. (2018), "Scientific Data Analysis Using Data-Intensive Scalable Computing: The SciDISC Project", LADaS 2018, p 1-8.

UDFs

- O uso de UDF traz desafios na otimização de execução
  - Ausência de semântica clara sobre seus comportamentos
  - Possuem códigos *ad hoc* relacionados ao domínio
  - *frameworks* como o Spark são incapazes de otimizar
- Rheinlander et al. (2017): Estudo extenso sobre otimização de execução de UDF em *dataflows*
  - Apontam principais desafios
  - Possíveis soluções
    - Álgebras de workflows: Ogasawara et al. 2011, Rheinlander et al. 2015, Fegaras 2017

Rheinländer, A., Leser, U., Graefe, G., (2017), "Optimization of complex dataflows with user-defined functions", *ACM Computing Surveys*, v. 50, n. 3.
Ogasawara, E., de Oliveira, D., Valduriez, P., Dias, J., Porto, F., Mattoso, M., (2011), "An algebraic approach for data-centric scientific workflows", *Proceedings of the VLDB Endowment*, v. 4, p. 1328–1339.
Rheinlander, A., Heise, A., Hueske, F., Leser, U., Naumann, F., (2015), "SOFA: An extensible logical optimizer for UDF-heavy data flows", *Information Systems*, v. 52, p. 96–125.
Fegaras, 2017, "An algebra for distributed Big Data Analytics", Journal of Functional Programming , p. 1–47

Álgebra de Workflows

- Relações como modelo para consumo e produção de dados
- Relação R possui um esquema $\mathcal{R}$
  - Tipos primitivos (int, float, String, date) e referências a arquivos (URI)
  - Atr(R) e Key(R)
  - Relações podem ser manipuladas:
    - União (∪), interseção (∩), diferença (−)
    - Atribuição a variáveis: $T \leftarrow R_1 \cup R_2$

Atividades

- UDFs que encapsulam a invocação de programas
- Estabelecem um esquema de relação para consumo e produção de dados
- Regidas por operações algébricas
  - Estabelecem relação de consumo e produção de dados

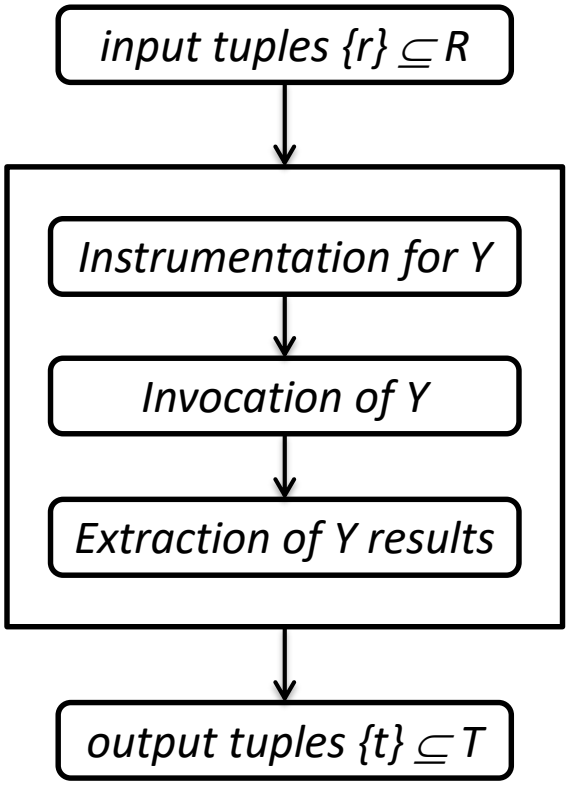
Operações da Álgebra de Workflows

Operação	UDF	Operandos adicionais	Resultado	Cons. $\times$ Prod. tuplas
Map	programa	Relação R	Relação S	$1:1$ *por* $\|R\|$
SplitMap	programa	Relação R	Relação S	$1:m$ *por* $\|R\|$
Reduce	programa	Relação R e atrs	Relação S	$n:1$ *por* $\|\pi_{atrs}\|$
Filter	programa	Relação R	Relação S	$1:(0-1)$ *por* $\|R\|$
SRQuery	exp. relacional	Relação R	Relação S	$n:m$
MRQuery	exp. relacional	Relações $(R_1 \cdots R_i)$	Relação S	$(n_1 \cdots n_i):m$

Formalização da otimização usando operações algébricas

Ativação de Atividade

Uma ativação é um objeto autocontido contendo toda a informação necessária para execução da atividade em uma unidade computacional (i.e. o programa a executar e o dados em sua granularidade mínima)



Bouganim, L., Florescu, D., Valduriez, P., (1996), "Dynamic Load Balancing in Hierarchical Parallel Database Systems". In: *Proceedings of the 22th International Conference on Very Large Data Bases*, p. 436–447, San Francisco, CA, USA.

Exemplo

Map

T

Dia	Trajectories
2015-01-01	2015-01-01.rdata
2015-01-02	2015-01-02.rdata

U ← Map(CalculaThresholds, T)

U

Dia	Trajectories	minvel	maxvel
2015-01-01	2015-01-01.rdata	0	80
2015-01-02	2015-01-02.rdata	0	96

Reduce

S

Dia	Seq	Trajetorias_min
2015-01-01	1	2015-01-01-00:00.json
2015-01-01	2	2015-01-01-00:01.json
…	…	….

T ← Reduce(ProduceRData, {'Dia'}, S)

T

Dia	Trajetorias
2015-01-01	2015-01-01.rdata
…	…

Split Map

R

Dia	Malha	Mobilidade
2015-01-01	500x500	2015-01-01.ZIP
2015-01-02	500x500	2015-01-02.ZIP
2015-01-03	500x500	2015-01-03.ZIP

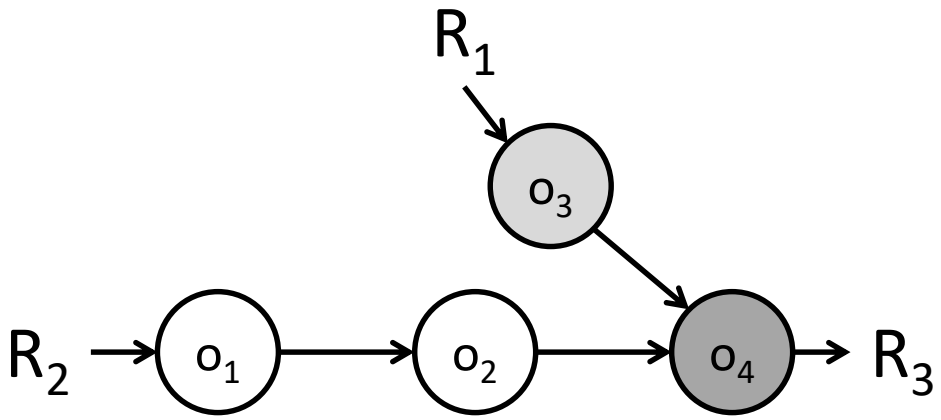
S ← SlipMap(extractTrajectories, R)

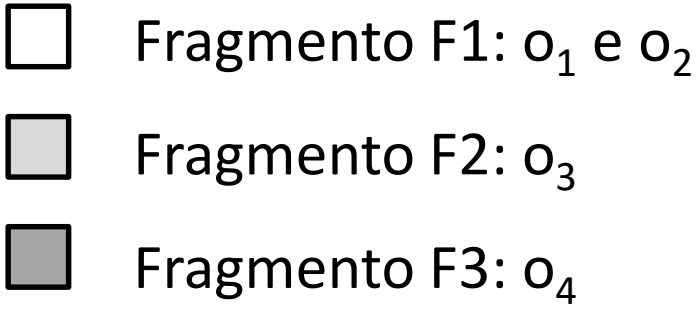
S

Dia	Seq	Trajetorias_min
2015-01-01	1	2015-01-01-00:00.json
2015-01-01	2	2015-01-01-00:01.json
…	…	….

Fragmento de Workflows

- Um fragmento *F* de um workflow W é um subconjunto de operações *o* de W tal que:
  - ou *F* é um conjunto unitário ou
  - $\forall o_j \in F,\ \exists o_i \in F \mid (Dep(o_i, o_j)) \vee (Dep(o_j, o_i))$

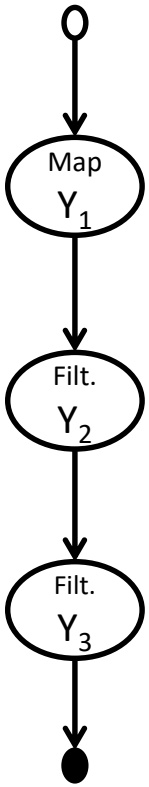




Modelo de Execução de Workflow

- Modelo depende da plataforma
  - Ativação da atividade
  - Estabelecimento de fragmentos de workflow
  - Estratégias de fluxo de dados e despacho das ativações
- Contexto de exemplo deste trabalho
  - “Distribuição estática” para despacho de ativações
  - Contém um conjunto de otimizações básicas para *dataflow* buscando pipelines (“fluxo de dados”) sempre que possível (“fragmentos de workflow”)
  - Cenário semelhante ao Apache Spark

Processo de otimização de workflow



(a)

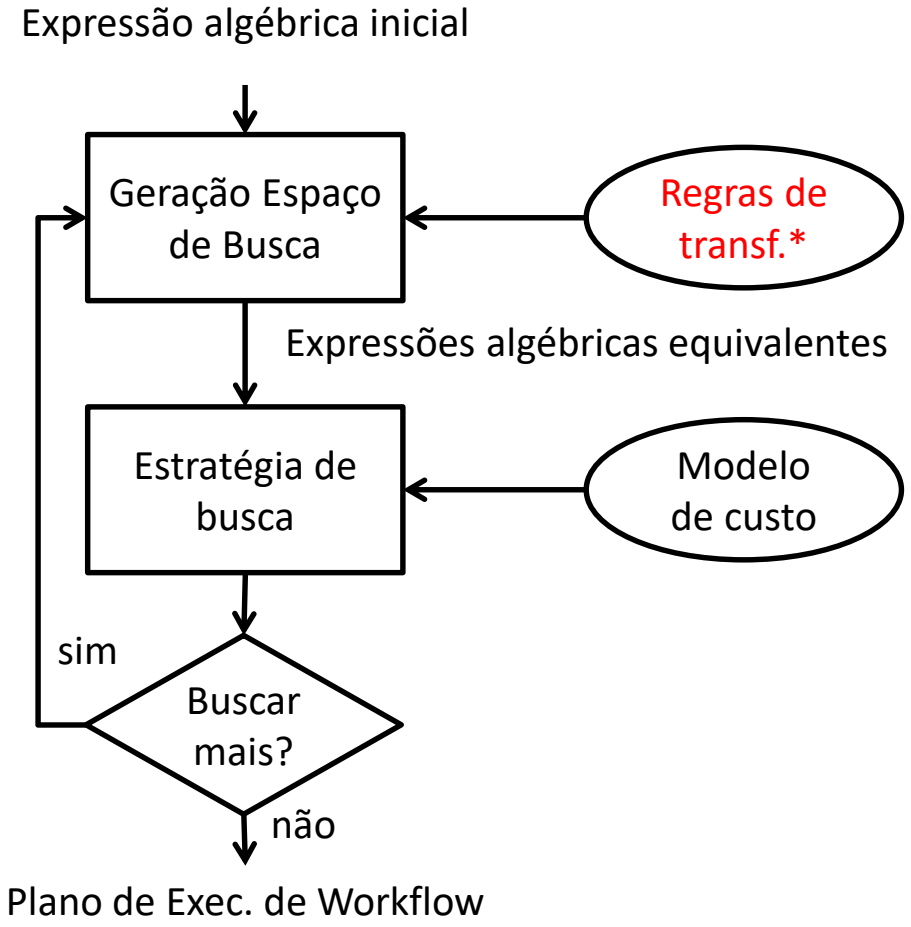
$$S \leftarrow Map(Y_1, R)$$

$$T \leftarrow Filter(Y_2, S)$$

$$U \leftarrow Filter(Y_3, T)$$

R(<u>id</u>, a, t)
S(<u>id</u>, a, b, t)
T(<u>id</u>, a, b, t)
U(<u>id</u>, a, b, t)

(b)



(c)

Oportunidade de integrar processamento de consulta relacional com álgebra de workflows

- Uniformização do modelo relacional
- Integração das transformações algébricas da álgebra relacional com álgebra de workflows
- Mapeamento da álgebra de workflows à álgebra relacional

Operação	Álgebra de Workflow	Álgebra Relacional
Map	$S \leftarrow Map(Y, R)$	$S \leftarrow \pi_{u_Y(*)} R$
SplitMap	$S \leftarrow SplitMap(Y, R)$	$S \leftarrow \overline{\pi}_{u_Y(*)} R$
Reduce	$S \leftarrow Reduce(Y, atrs, R)$	$S \leftarrow {}_{atrs}\Gamma_{u_Y(*)} R$
Filter	$S \leftarrow Filter(Y, R)$	$S \leftarrow \sigma_{u_Y(*)} R$

Estudo de caso

- Álgebra de workflows
  - $S \leftarrow Map(Y_1, R)$
  - $T \leftarrow Map(Y_2, S)$
  - $U \leftarrow Map(Y_3, T)$
- Tradução para álgebra relacional
  - $S \leftarrow \pi_{id,a,b,t}\ (\pi_{uY_1(id,a,t)} R)$
  - $T \leftarrow \pi_{id,a,b,t}\ (\sigma_{uY_2(id,a,b,t)} S)$
  - $U \leftarrow \pi_{id,a,b,t}\ (\sigma_{uY_3(id,a,b,t)} T)$
- Reescrita
  - $U \leftarrow \pi_{id,a,b,t}\left(\sigma_{uY_3(id,a,b,t)} \pi_{id,a,b,t}\ (\sigma_{uY_2(id,a,b,t)} S)\right)$
- Comutatividade
  - $U \leftarrow \pi_{id,a,b,t}\left(\sigma_{uY_2(id,a,b,t)} \pi_{id,a,b,t}\ (\sigma_{uY_3(id,a,b,t)} S)\right)$
  - $W \leftarrow \pi_{id,a,b,t}\ (\sigma_{uY_3(id,a,b,t)} S)$
  - $U \leftarrow \pi_{id,a,b,t}\ (\sigma_{uY_2(id,a,b,t)} W)$

Considerações finais

- Considere custo computacional para Y_1, Y_2 e Y_3 respectivamente, de 1, 4 e 0.25.
- Considere que seletividade de Y_2 seja 100% (todas as tuplas são aceitas) e custo computacional por tupla Y_3 variando o valor de seletividade de 20% a 100% em múltiplas execuções do workflow
  - A versão otimizada trouxe melhoria de 60% na execução com em um cluster de 64 núcleos
- Próximos passos
  - Formalizar de modo mais rigoroso o comportamento de expressões algébricas
    - Explicitar transformações
    - Provas de correção
    - Atributos consumíveis

Contato

CEFET/RJ

Eduardo Ogasawara
eogasawara@cefet-rj.br
http://eic.cefet-rj.br/~eogasawara

Agradecimentos