Introdução à Programação com Swift: Workflows Científicos Paralelos

Professores: Luiz Gadelha, Kary Ocaña

Programa de Verão do LNCC, 2017
Laboratório Nacional de Computação Científica

10 de abril de 2018

Swift

- Swift (`http://www.swift-lang.org`) permite gerenciar workflows científicos em ambientes paralelos e distribuídos.
- Usabilidade e produtividade com abstrações de alto nível para especificação e paralelização de workflows científicos.
- Alta taxa de execução de tarefas (500 tps no Swift/K e 500 Ktps no Swift/T).

M. Wilde, M. Hategan, J. Wozniak, B. Clifford, D. Katz, and I. Foster. Swift: A language for distributed parallel scripting. *Parallel Computing*, 37(9):634-652, 2011.

Wozniak, J. M. et al. (2013). Swift/T: Large-Scale Application Composition via Distributed-Memory Dataflow Processing. Proc. 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (CCGrid 2013), pp. 95–102.

Armstrong, T. G. et al. (2014). Compiler Techniques for Massively Scalable Implicit Task Parallelism. In Proc. International Conference for High Performance Computing, Networking, Storage and Analysis (SC 2014), pp. 299–310.

Swift: Gerenciamento de Proveniência

- Durante o *Third Provenance Challenge* apresentamos um modelo de proveniência retrospectiva para MTC e uma implementação no Swift.
- Problemas detectados:
  - Ausência de informações do domínio científico dificultava a interpretação dos dados de proveniência.
  - Dificuldade de escrita de consultas no modelo relacional: junções e fecho transitivo.

L. Gadelha, B. Clifford, M. Mattoso, M. Wilde, and I. Foster. Provenance Management in Swift. *Future Generation Computer Systems*, 27(6):775-780, 2011.

Swift: Gerenciamento de Proveniência

- MTCProv é uma ferramenta de proveniência para workflows paralelos e distribuídos integrada ao Swift.
- Metodologia seguida:
  1. Levantamento de padrões de consulta.
  2. Modelagem de dados.
  3. Implementação de mecanismos de coleta.
  4. Implementação de interface de consulta.

MTCProv: Padrões de Consultas

- Além das informações estruturais do grafo de proveniência, um cientista quer saber:
  - Como avaliar o resultado de uma execução de workflow científico?
  - Qual o grau de precisão e confiança de uma simulação?
  - Qual foi custo computacional para se chegar ao resultado?
  - A computação pode ser reproduzida para verificação e confirmação de um resultado?
  - Qual a correlação entre performance científica e computacional?

L. Gadelha, M. Wilde, M. Mattoso, I. Foster. Exploring provenance in high performance scientific computing. *Proceedings of the First Annual Workshop on High Performance Computing Meets Databases* (HPCDB 2011), pp. 17–20. ACM, 2011.

MTCProv: Padrões de Consulta

- Padrões:
  - Atributo de entidade (EA).
  - Relacionamento direto (R).
  - Relacionamento transitivo (R^*).
  - Casamento de grafos (LGM).
  - Resumo de execução (RS).
    - Performance computacional (RRP).
    - Performance científica (RSP).
  - Comparação entre execuções (RCp).
  - Correlações entre execuções (RCr).

L. Gadelha, M. Mattoso, M. Wilde, I. Foster, Provenance Query Patterns for Many-Task Scientific Computations. *Proceedings of the 3rd USENIX Workshop on Theory and Applications of Provenance* (TaPP'11), 2011.

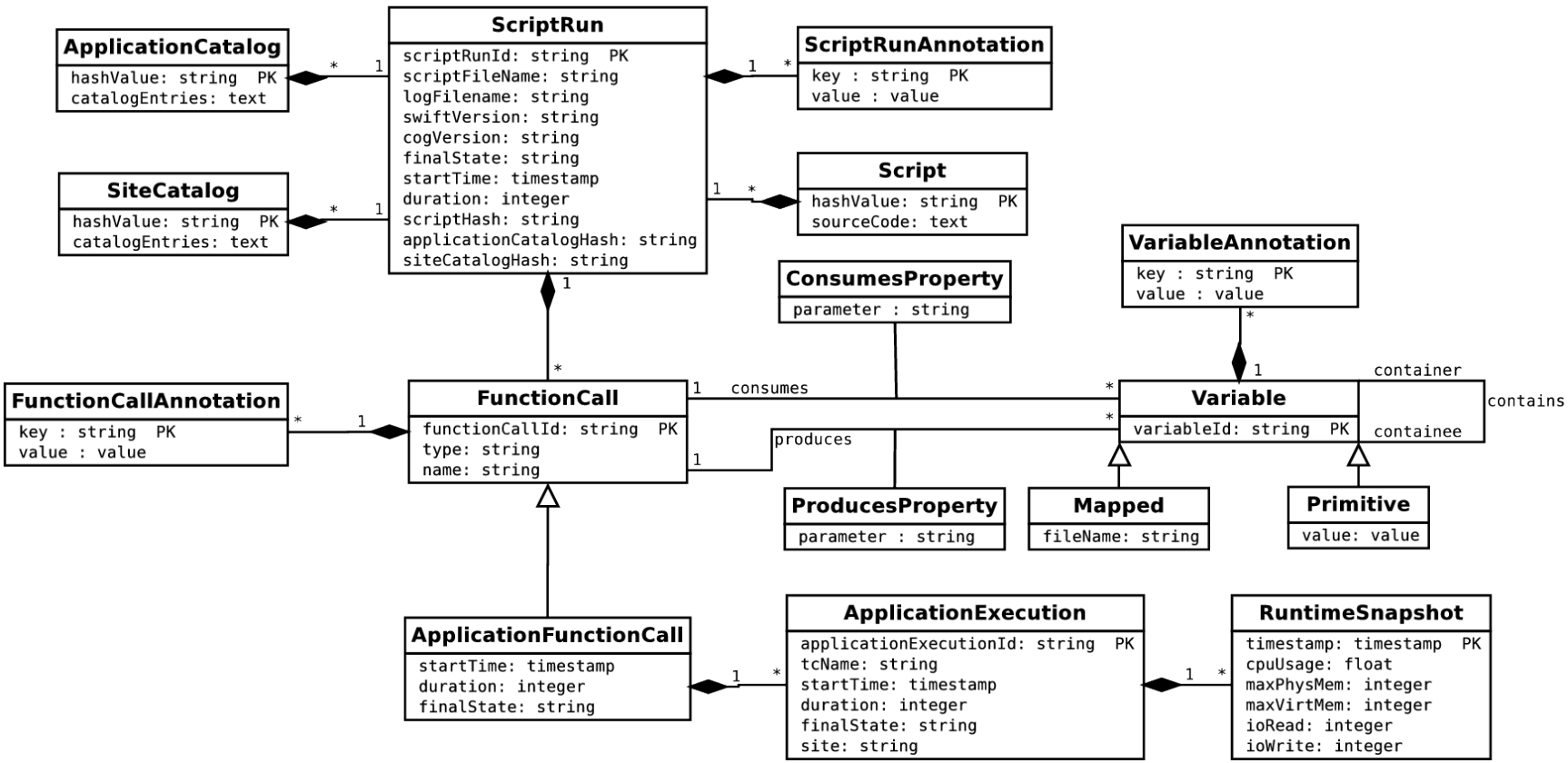
MTCProv: Padrões dos *Provenance Challenges*

Padrão	PC1/PC2									PC3				PC3 (Consultas Opcionais)													
	1	2	3	4	5	6	7	8	9	1	2	3	5	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
EA	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
R	×	×	×		×	×	×	×	×	×	×	×	×		×	×	×	×	×	×	×	×	×	×	×	×	×	×
R*	×	×	×			×	×			×	×	×	×		×		×				×	×			×	×	×	×
LGM							×								×													
RS	×	×	×							×	×	×	×	×		×	×						×	×	×	×	×	
RCp					×	×	×	×	×						×			×	×	×	×	×						×
RCr				×																								

MTCProv: Modelagem de Dados

- Objetivos:
  1. Coletar informações sobre consumo e produção de artefatos e processos.
  2. Coletar informações sobre hierarquia entre artefatos de dados.
  3. Permitir o enriquecimento das informações de proveniência com anotações.
  4. Coletar informações sobre versionamento de workflows científicos e aplicações componentes.
  5. Coletar informações de tempo de execução das aplicações componentes.
  6. Prover uma interface de consulta amigável e útil para as informações de proveniência.

Swift: Gerenciamento de Proveniência



L. Gadelha, M. Wilde, M. Mattoso, and I. Foster. MTCProv: a practical provenance query framework for many-task scientific computing. *Distributed and Parallel Databases*. Springer, 2012.

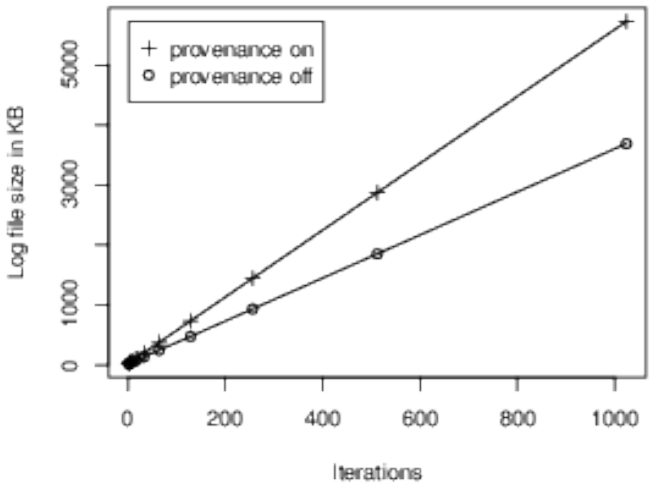
MTCProv: Captura e Armazenamento

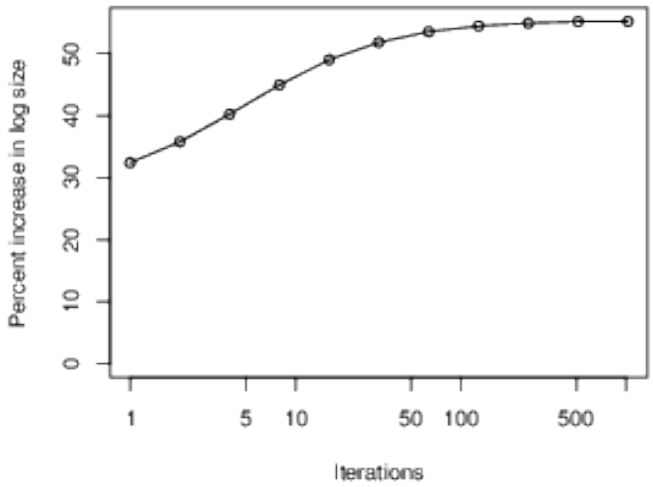
- Informações de proveniência são extraídas dos logs gerados Swift a cada execução de script.
- Armazenamento em banco de dados relacional:
  - As restrições de se utilizar um esquema de dados fixo podem ser reduzidas com o uso de anotações.
  - Fecho transitivo pode ser calculado facilmente com funções recursivas nativas a partir do SQL:1999.
- Modelos de dados baseados em grafos requerem travessia frequente para recuperação de atributos em consultas de agregação.

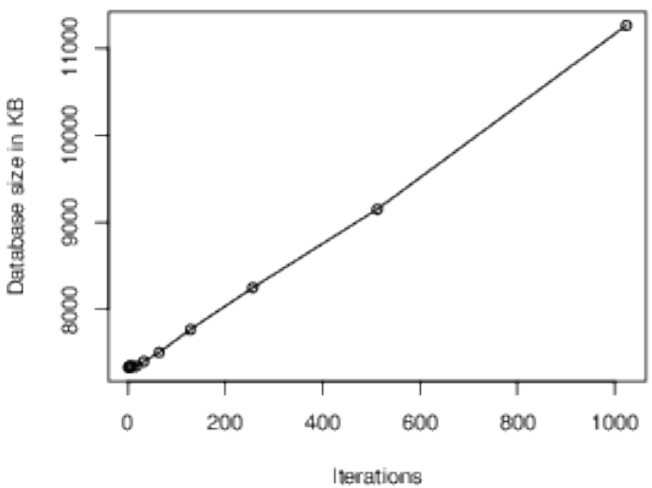
MTCProv: Captura e Armazenamento

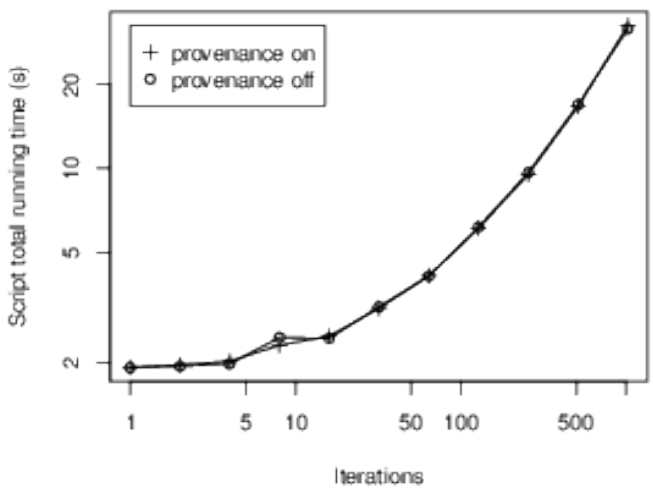
- Anotações podem ser coletadas por:
  - scripts *ad hoc* para um workflow científico,
  - scripts que iniciam a execução remota (*wrapper scripts*).
- Exemplos de anotações:
  - Parâmetros científicos não visíveis ao Swift.
  - Informações sobre segurança.

MTCProv: Impacto da Captura e Armazenamento



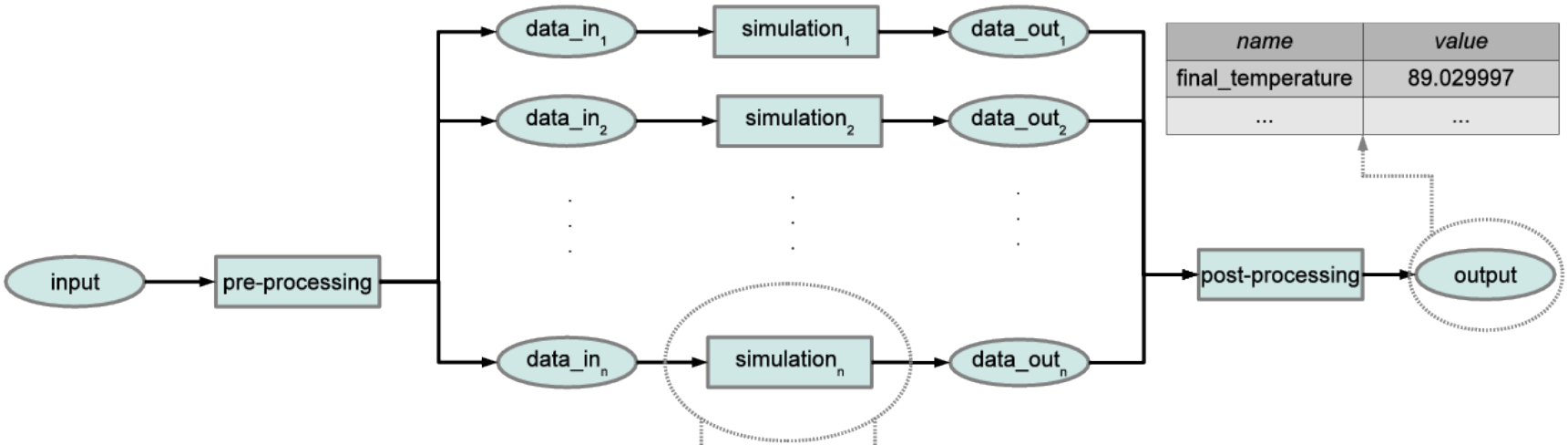






Laboratório Nacional de Computação Científica

MTCProv: Captura e Armazenamento de Anotações



name	value
final_temperature	89.029997
...	...

name	value
execution_site	pads.ci.uchicago.edu
version	1.15
...	...

timestamp	mem_usage	cpu_load	io_read	io_write
1314104905	126342	99	230	0
1314104906	155739	100	123	1044
...	...	...	...	...
1314112105	394523	99	0	129

L. Gadelha, M. Wilde, M. Mattoso, and I. Foster. Exploring provenance in high performance scientific computing. Proc. Workshop on High Performance Computing Meets Databases (HPCDC'11), pp. 17–20, 2011.

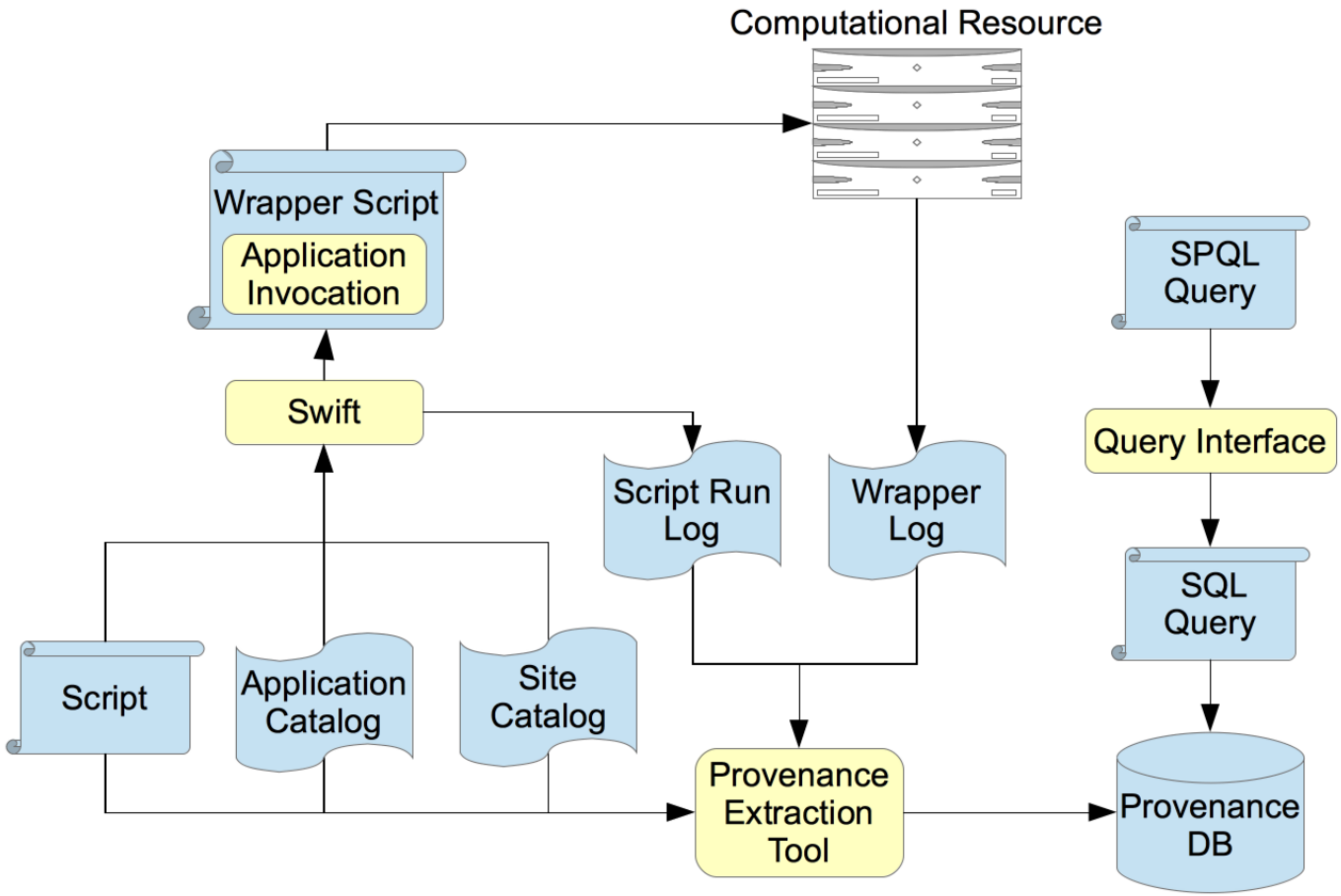
MTCProv: Interface de Consulta

- Abstração de padrões de consultas comuns através de funções e procedimentos SQL.
- O padrão R^* é implementado com funções recursivas nativas do SQL no procedimento `ancestors`.
- Os padrões RCp e RCr são implementados pelo procedimento:
  - `compare_run`($\langle$ *lista de parâmetros* ou *chaves de anotações*$\rangle$) retorna uma tabela com os valores das anotações e parâmetros por execução de script.
- Foi implementada uma interface de consultas em Java/ANTLR que calcula automaticamente cláusulas `FROM` e expressões para junções.

Swift: Gerenciamento de Proveniência

- Informações de proveniência são extraídas dos logs gerados Swift a cada execução de script.
- Armazenamento em banco de dados relacional:
  - As restrições de se utilizar um esquema de dados fixo podem ser reduzidas com o uso de anotações.
  - Fecho transitivo pode ser calculado facilmente com funções recursivas nativas a partir do SQL:1999.
- Modelos de dados baseados em grafos requerem travessia frequente para recuperação de atributos em consultas de agregação.

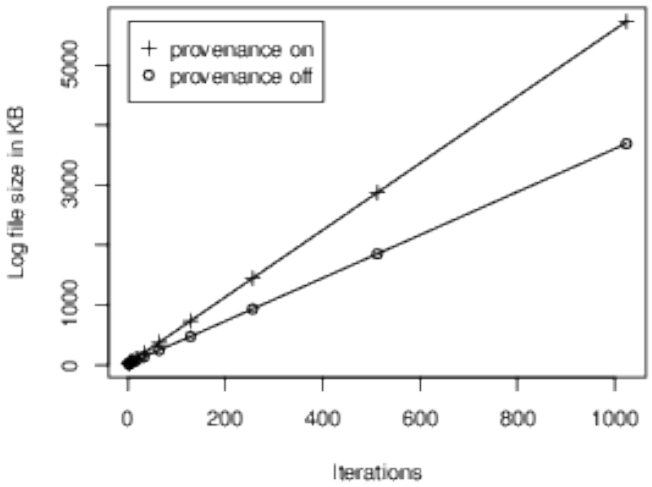
Swift: Gerenciamento de Proveniência

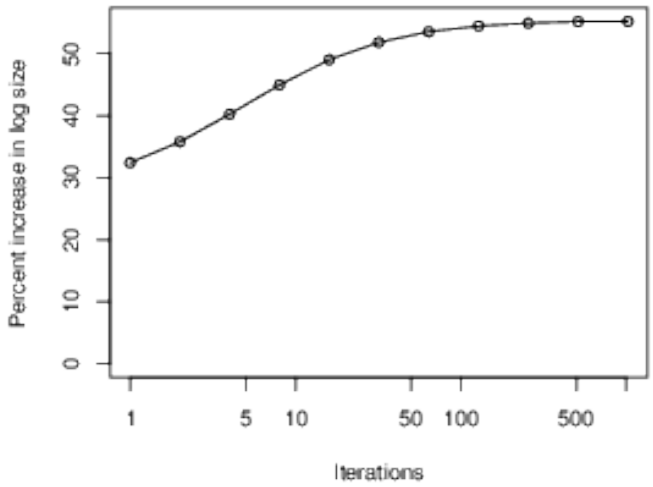


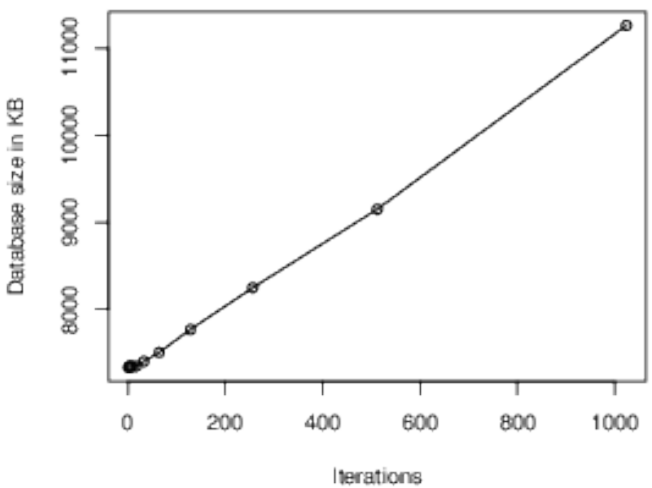
Swift: Gerenciamento de Proveniência

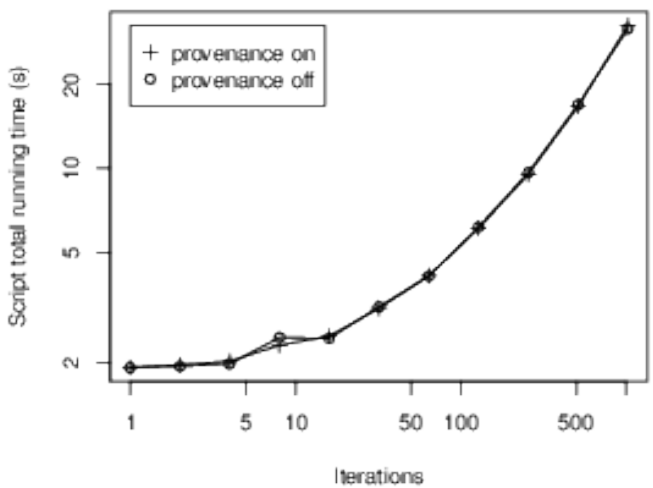
- Anotações podem ser coletadas por:
  - scripts *ad hoc* para um workflow científico,
  - scripts que iniciam a execução remota (*wrapper scripts*).
- Exemplos de anotações:
  - Parâmetros científicos não visíveis ao Swift.
  - Informações sobre segurança.

Swift: Gerenciamento de Proveniência



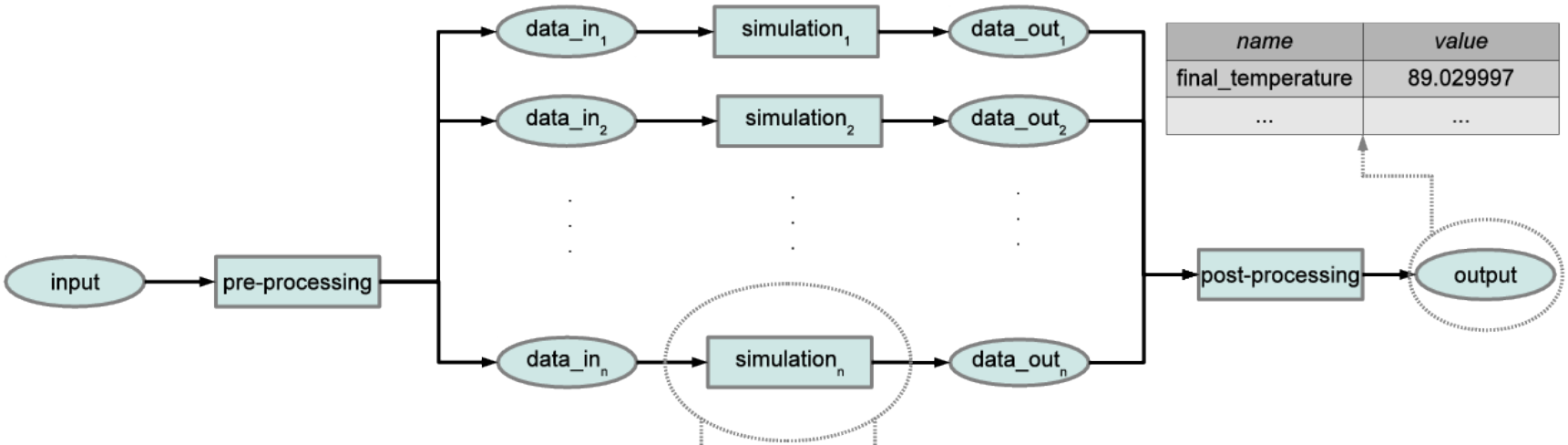






Laboratório Nacional de Computação Científica

Swift: Gerenciamento de Proveniência



name	value
final_temperature	89.029997
...	...

name	value
execution_site	pads.ci.uchicago.edu
version	1.15
...	...

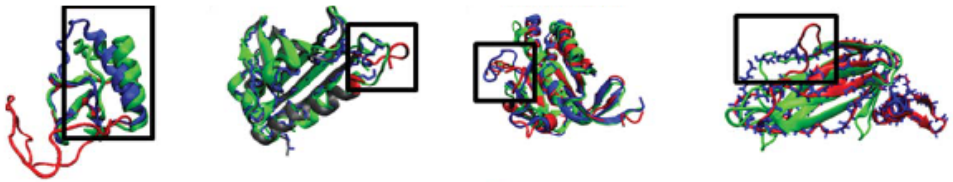
timestamp	mem_usage	cpu_load	io_read	io_write
1314104905	126342	99	230	0
1314104906	155739	100	123	1044
...	...	...	...	...
1314112105	394523	99	0	129

L. Gadelha, M. Wilde, M. Mattoso, and I. Foster. Exploring provenance in high performance scientific computing. Proc. Workshop on High Performance Computing Meets Databases (HPCDC'11), pp. 17–20, 2011.

Swift: Gerenciamento de Proveniência

- Abstração de padrões de consultas comuns através de funções e procedimentos SQL.
- O padrão R* é implementado com funções recursivas nativas do SQL no procedimento `ancestors`.
- Os padrões RCp e RCr são implementados pelo procedimento:
  - `compare_run`(⟨ *lista de parâmetros* ou *chaves de anotações*⟩) retorna uma tabela com os valores das anotações e parâmetros por execução de script.
- Foi implementada uma interface de consultas em Java/ANTLR que calcula automaticamente cláusulas `FROM` e expressões para junções.

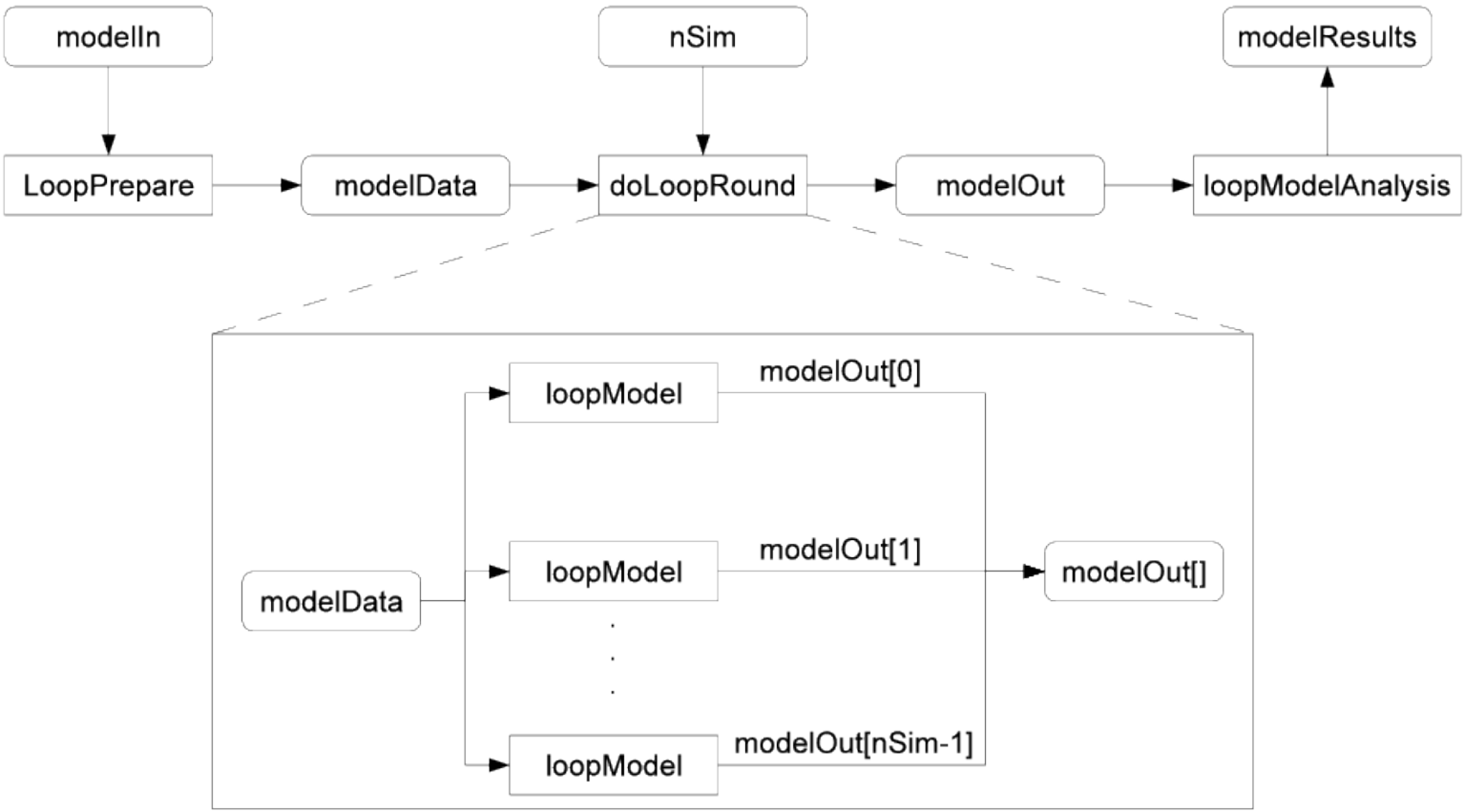
Estudo de Caso: Open Protein Simulator



- Open Protein Simulator (OOPS) é uma aplicação para modelagem de estrutura de proteínas.
- Informação visível ao Swift:
  - Parâmetros científicos (p.ex. identificador da proteína).
  - Informações do tempo de execução (p.ex. duração de execuções).
- Um script coleta informações adicionais:
  - Parâmetros científicos não visíveis para o Swift.
  - Versões SVN das aplicações componentes e do script Swift.

A. Adhikari, J. Peng, M. Wilde, J. Xu, K. Freed, and T. Sosnick, Modeling large regions in proteins: Applications to loops, termini, and folding. *Protein Science* 21(1):107–121, 2012.

Estudo de Caso: Open Protein Simulator



Estudo de Caso: Open Protein Simulator

Listar execuções entre duas datas:

```
select script_run
where  script_run.start_time between '2010-04-04' and '2010-08-08' and
       script_run.filename='psim.loops.swift';

                id                 |         start_time          |   ...
-----------------------------------+-----------------------------+--------
 psim.loops-20100619-0339-b95ull7d | 2010-06-19 03:39:15.18-05   |   ...
 psim.loops-20100618-0402-qhm9ugg4 | 2010-06-18 04:02:21.234-05  |   ...
              ...                  |             ...             |   ...
```

Estudo de Caso: Open Protein Simulator

Correlação entre número de simulações e RMSD (performance científica):

```
SELECT run_id, r.value as nSim, t.value as rmsd
FROM   compare_run_by_param('proteinId') as r
       INNER JOIN
       compare_run_by_param('nSim') as s USING (run_id)
       INNER JOIN
       compare_run_by_annot('rmsd') as t USING (run_id)
WHERE  r.value='TR567' and run.id LIKE 'psim.loops%';
```

```
             run_id                | nSim |  rmsd
-----------------------------------+------+---------
 psim.loops-20100604-2215-cdifsnb3 |  256 | 3.33123
 psim.loops-20100613-0125-keyyyc35 |  512 | 0.76274
 psim.loops-20100616-1512-h6q4g4ja | 1024 | 0.68426
 ...
```

Swift: Gerenciamento de Proveniência

- Implementação atual baseada em um esquema de dados simplificado.
- `swiftlog -import-provenance` ⟨dir execução⟩.
- Gera banco de dados de proveniência no SQLite.
- Ferramenta HPSW-Prof, análise de proveniência e predições baseadas em aprendizagem de máquina.

Mondelli, M. L., de Souza, M. T., Ocaña, K., Vasconcelos, A. T. R. de, Gadelha Jr, L. M. R. (2016). HPSW-Prof: A Provenance-Based Framework for Profiling High Performance Scientific Workflows. In Proc. of Satellite Events of the 31st Brazilian Symposium on Databases (SBBD 2016) (pp. 117–122). SBC.

Swift: Laboratório

- Tutorial Swift:
  - `http://www.swift-lang.org/swift-tutorial/doc/tutorial.html`

Swift: Laboratório

- BLAST Paralelo:
  - `git clone https://github.com/lgadelha/swift-blast`
  - `wget http://buriti.lncc.br/genes.pep`
  - `wget http://buriti.lncc.br/sequence.seq`
  - `wget http://buriti.lncc.br/parallelblast_2.0.9.tar.gz`

Obrigado!

E-mail: lgadelha@lncc.br