Реляционная интерпретация многопоточных языков программирования

Евгений Моисеенко
научный руководитель Д.Ю.Булычев
рецензент В.А.Павлов

22 июля 2018 г.

Функциональное программирование

Программы представляются как функции

$$\text{sort} :: \text{int list} \to \text{int list}$$

$$[2; 1; 3] \longrightarrow [1; 2; 3]$$

Реляционное программирование

Программы представляются как отношения

$$\text{sort} :: \text{int list} \times \text{int list}$$

$$\begin{array}{l} [1;2;3] \\ [1;3;2] \\ [2;1;3] \\ [2;3;1] \\ [3;1;2] \\ [3;2;1] \end{array} \qquad \longleftrightarrow \qquad [1;2;3]$$

Реляционный интерпретатор

Семантика языка представляется как отношение

- Исполнение
- Верификация
- Синтез

Задачи

1. Разработать реляционный интерпретатор (на языке OCanren) для императивного многопоточного языка
2. Разработать необходимые расширения для языка OCanren
3. Провести апробацию реляционного интерпретатора

Модели памяти

$$\begin{array}{l} [x] := 1; \\ [f] := 1; \end{array} \;\Big\|\; \begin{array}{l} r_1 := [f]; \\ r_2 := [x]; \end{array}$$

Поведение $r_1 = 1 \wedge r_2 = 0$:

- невозможно наблюдать в модели последовательной консистентности (SC)
- можно наблюдать в более слабых моделях

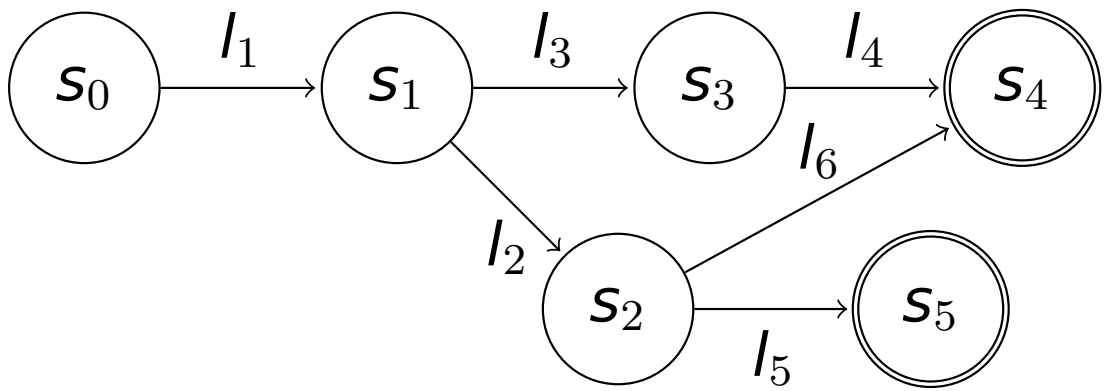
Модели памяти

- Интерпретатор состоит из двух подсистем
  - Подсистема потоков
  - Подсистема памяти

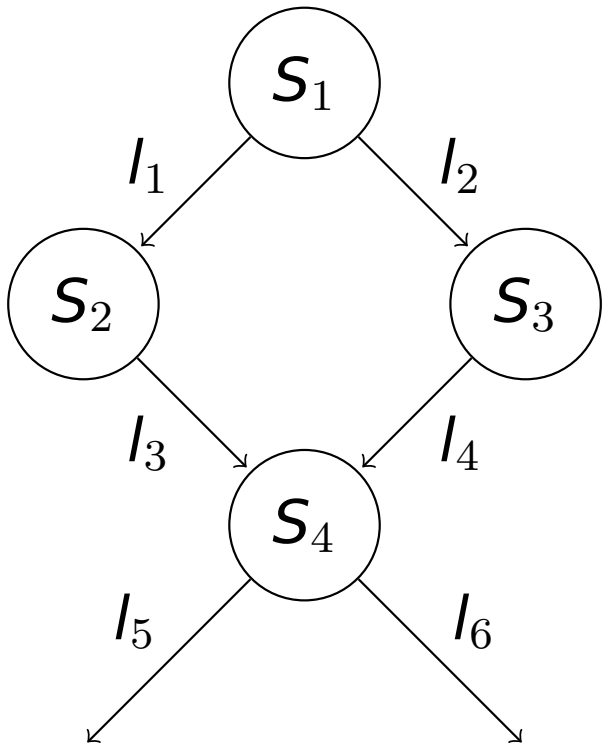
- Поддержаны модели памяти
  - SC — Sequential Consistency
  - TSO (x86) — Total Store Ordering
  - SRA — Strong Release-Acquire

Интерпретатор

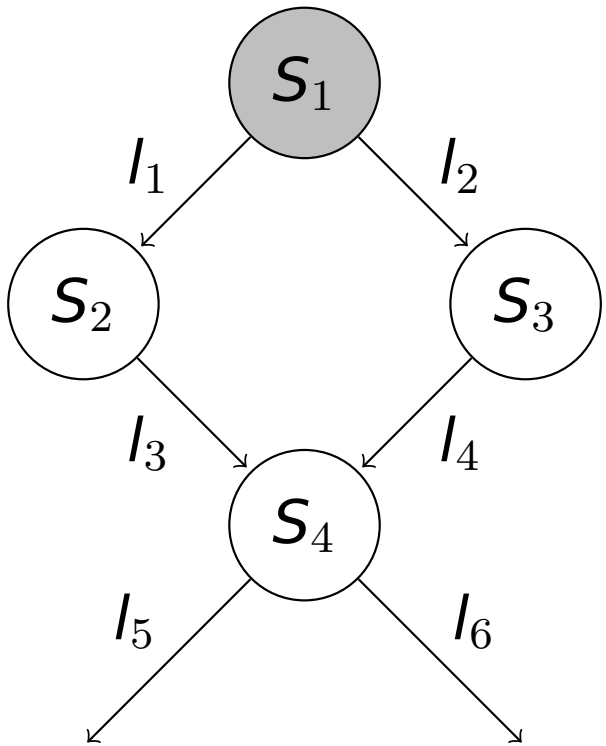
- Системы помеченных переходов
  - $s \xrightarrow{l} s'$
  - $s \rightarrow^* s'$



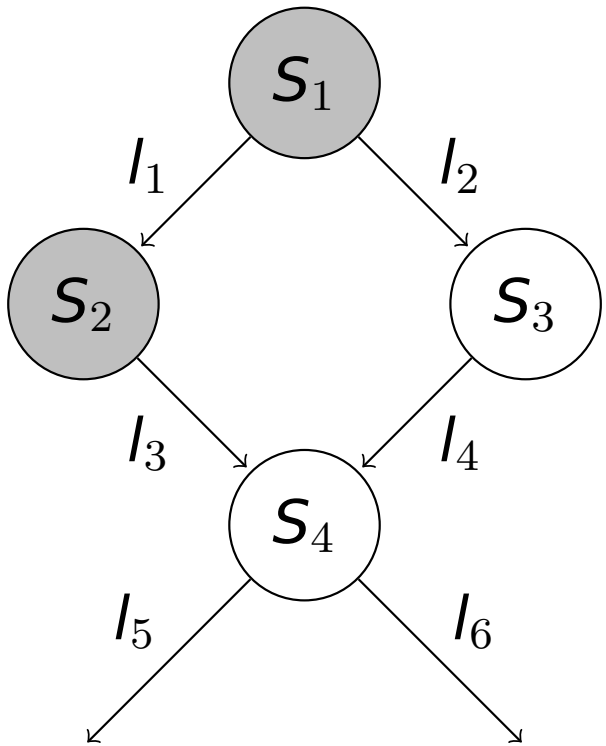
Обход пространства состояний



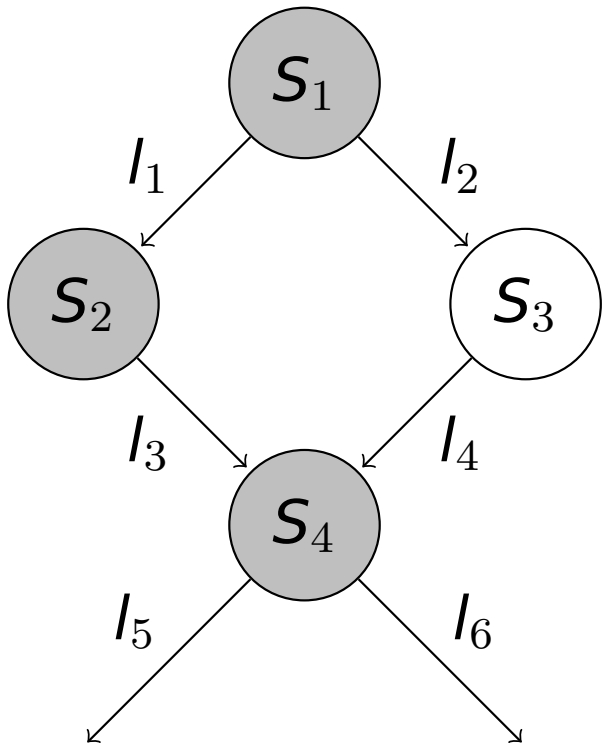
Обход пространства состояний



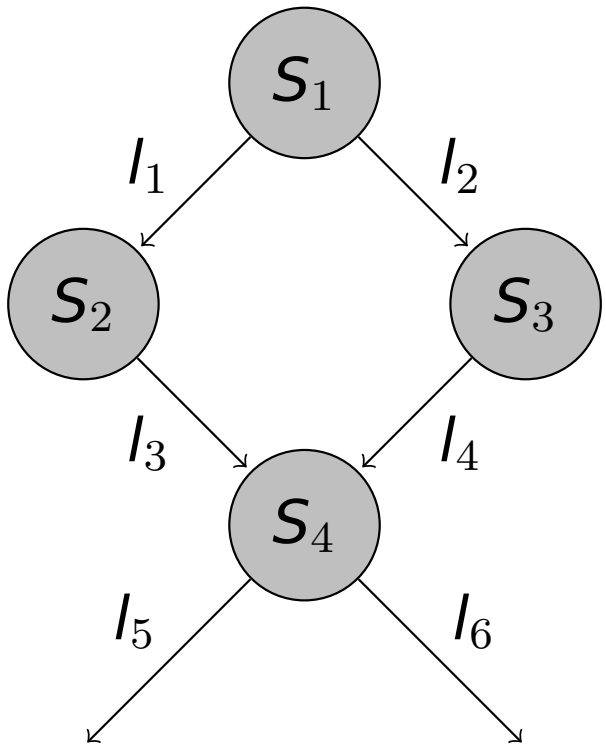
Обход пространства состояний



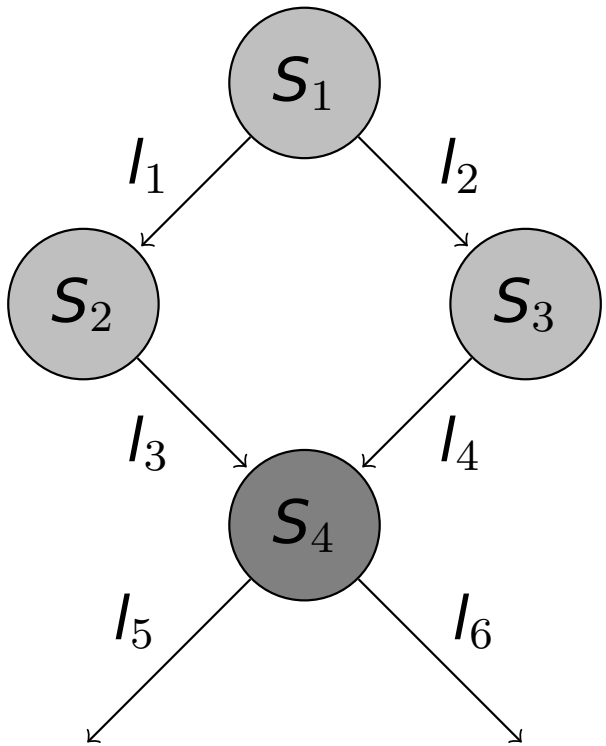
Обход пространства состояний



Обход пространства состояний



Обход пространства состояний



Табличная мемоизация

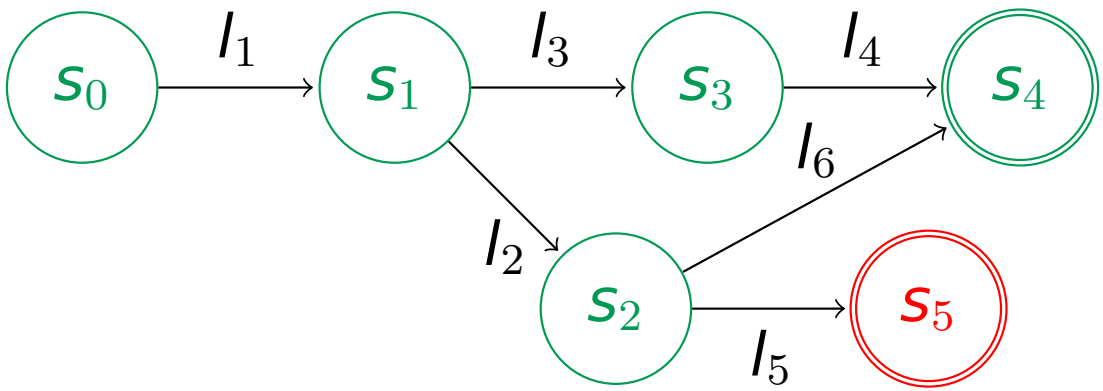
- Сохраняем и переиспользуем результаты запросов

- С каждым отношением связываем таблицу
  - Ключ — запрос
  - Значение — список ответов

Проверка инвариантов

$$\forall s.(s_{init} \rightarrow^* s) \Rightarrow safe(s)$$

$$\neg\exists s.(s_{init} \rightarrow^* s) \wedge \neg safe(s)$$



$$s_0 \rightarrow^* s_5 \wedge \neg safe(s_5)$$

Конструктивное стратифицированное отрицание

Исполняем запрос и строим на основе ответов набор ограничений

Потребовалось добавить поддержку двух новых типов ограничений

- $\forall \overline{x}.t \not\equiv u$
- $\forall \overline{x} \exists \overline{y}.t \equiv u$

Апробация: лакмусовые тесты

Название теста	SC	TSO	SRA
Store Buffering		✓	
Store Buffering (+sc)		✓	
Load Buffering		✓	
Message Passing		✓	
Message Passing (+rel+acq)		✓	
Coherence of Read-Read		✓	
Independent Reads of Independent Writes		✓	
Write-to-Read Causality		✓	

Апробация: верификация

$$\begin{array}{l} [\mathsf{x}] := 1; \\ [\mathsf{f}] := 1; \end{array} \Bigg\| \begin{array}{l} \mathsf{r}_1 := [\mathsf{f}]; \\ \mathsf{r}_2 := [\mathsf{x}]; \end{array}$$

$s_{init} \equiv \langle prog_{MP}, mem_{init} \rangle$

$\neg \exists s.\ eval_{SC}(s_{init}, s) \wedge s.r_1 = 1 \wedge s.r_2 = 0$

Апробация: верификация

Название теста	SC	TSO	SRA
Message Passing	0.01	0.02	0.02
Barrier	0.08	0.14	0.12
Cohen Lock	0.11	0.35	0.21
Dekker Lock	0.11	1.08	0.21
Peterson Lock	0.26	2.91	0.48

Апробация: генерация модификаторов доступа

$$\begin{array}{l|l|l} [\mathsf{x}]_{na} := 1; & & \mathsf{r}_1 := [\mathsf{f}]_r; \\ [\mathsf{f}]_q := 1; & & \mathsf{r}_2 := [\mathsf{x}]_{na}; \end{array}$$

$s_{init} \equiv \langle prog_{MP}, mem_{init} \rangle$

$q, r : \neg\exists s.\ eval_{SRA}(s_{init}, s) \wedge s.r_1 = 1 \wedge s.r_2 = 0$

$q \mapsto rel; r \mapsto acq$

Апробация: генерация модификаторов доступа

Название теста	TSO	SRA
Message Passing	0.05	0.36
Barrier	2.39	238.67
Cohen Lock	1.00	2.48
Dekker Lock	76.49	OOM[1]
Peterson Lock	5300.51	OOM[1]

[1] Out of memory

Апробация: выводы

- Интерпретатор применим для исполнения и верификации программ

- Интерпретатор значительно хуже справляется с задачами синтеза

Результаты

1. Разработан интерпретатор для императивного многопоточного языка
2. Разработаны расширения OCanren
3. Проведена апробация