Notatki do pracy magisterskiej

Bartłomiej Królikowski

February 2024

1 Cel

Weryfikacja algorytmu Radix Heap w Coqu
(link: https://ocw.mit.edu/courses/15-082j-network-optimization-fall-2010/6852db65d4a72816ee86a35e0e546669_MIT15_082JF10_lec06.pdf)

2 O algorytmie

Algorytm jest modyfikacją algorytmu Dijkstry, w której korzysta się ze specjalnie zoptymalizowanej struktury danych, co pozwala zmniejszyć (zamortyzowaną) złożoność algorytmu, zachowując dobrą złożoność pamięciową.

Jego weryfikacja jest ciekawym problemem z trzech powodów:

- wykorzystana struktura danych jest dość skomplikowana i poprawne zaimplementowanie działających na niej operacji wymaga uwagi
- liczymy zamortyzowaną złożoność czasową
- liczymy zamortyzowaną złożoność pamięciową

Według mojej obecnej wiedzy weryfikacja tego algorytmu nie została jeszcze przeprowadzona w Coqu.

3 Sposób weryfikacji

Typowym sposobem weryfikacji algorytmu w Coqu jest zapisanie go jako obliczalnej funkcji w języku Coqa i później zapisywanie i dowodzenie różnych zdań logicznych łączące argumenty z wynikiem i czasem działania, o który różnymi metodami uzupełniane są obliczenia. Uważam, że to jest błąd, bo kod, który produkujemy, staje się przez to bardzo skomplikowany (a czasami musimy wręcz skorzystać z narzędzi zewnętrznych, których Coq nie weryfikuje) a więc mniej wiarygodny.

Uważam, że nie ma potrzeby by relacja między argumentami a wynikiem i kosztem (czasowym i pamięciowym) była obliczalną funkcją. Rezygnacja z tego warunku daje nam dużo możliwości uproszczenia metody, a nawet przeprowadzenia całości wewnątrz systemu Coq.

Mój pomysł polega na stworzeniu prostego języka z mutowalnymi referencjami (potrzebne w przypadku omawianego algorytmu, a przynajmniej przydatny jest mutowalny stan; spodziewam się, że będzie to pozwalało na śledzenie wykorzystania pamięci (być może po dodaniu regionów)) i stworzeniu dla niego semantyki kosztów.

W języku tym zapisywany będzie weryfikowany algorytm. Dzięki odpowiednio zdefiniowanym notacjom napis ten będzie czytany i tłumaczony do drzewa składniowego (zdefiniowanego jako konstrukcja w Coq) bezpośrednio przez system Coq. Otypowanie wyrażenia i dowód poprawności typu musiałyby być przeprowadzone ręcznie. Zasadniczą częścią dowodu stanowiłyoby definiowanie i dowodzenie spełnienia predykatów łączących wejście i wyjście algorytmu oraz jego koszt.

4 Rachunek z referencjami

Poniżej przedstawiam rachunek lambda z referencjami, który wymyśliłem na potrzeby opisanego zadania. Nie mam pewności czy jest to optymalne rozwiązanie. Nie udało mi się jeszcze udowodnić jego poprawności (własności bezpieczeństwa typów)

Gramatyka: (mamy zbiór L etykiet (Label, $l \in L$))

$$\mathsf{Val} \ni v ::= x \mid \langle\rangle \mid \lambda x.e \mid l \qquad \text{(values)}$$

$$\mathsf{Expr} \ni e ::= v \mid e\ e \mid \mathtt{deref}\ e \mid \mathtt{ref}\ e \mid e \leftarrow e \mid e; e \qquad \text{(expressions)}$$

$$\mathsf{Type} \ni \tau ::= \mathtt{U} \mid \tau \to \tau \mid \mathtt{R}\ \tau \qquad \text{(types)}$$

Reguły kontrakcji (σ oznacza stan abstrakcyjnej pamięci):

$$\frac{}{\langle(\lambda x.e)\ v, \sigma\rangle \rightharpoonup \langle e\{v/x\}, \sigma\rangle} \qquad \frac{l \notin \mathtt{dom}(\sigma)}{\langle \mathtt{ref}\ v, \sigma\rangle \rightharpoonup \langle l, \sigma[l \mapsto v]\rangle}$$

$$\frac{l \in \mathtt{dom}(\sigma)}{\langle \mathtt{deref}\ l, \sigma\rangle \rightharpoonup \langle \sigma(l), \sigma\rangle} \qquad \frac{l \in \mathtt{dom}(\sigma)}{\langle l \leftarrow v, \sigma\rangle \rightharpoonup \langle \langle\rangle, \sigma[l \mapsto v]\rangle} \qquad \frac{}{\langle \langle\rangle; v, \sigma\rangle \rightharpoonup \langle v, \sigma\rangle}$$

Reguły typowania:

$$\frac{}{\Gamma \vdash \langle\rangle : \mathtt{U}} \qquad \frac{(x:\tau) \in \Gamma}{\Gamma \vdash x : \tau} \qquad \frac{\Gamma, x : \tau_1 \vdash e : \tau_2}{\Gamma \vdash \lambda x.e : \tau_1 \to \tau_2}$$

$$\frac{\Gamma \vdash e_1 : \tau_2 \to \tau_1 \qquad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash e_1\ e_2 : \tau_1} \qquad \frac{\Gamma \vdash e : \tau}{\Gamma \vdash \mathtt{ref}\ e : \mathtt{R}\ \tau} \qquad \frac{\Gamma \vdash e : \mathtt{R}\ \tau}{\Gamma \vdash \mathtt{deref}\ e : \tau}$$

$$\frac{\Gamma \vdash e_1 : \mathtt{R}\ \tau \qquad \Gamma \vdash e_2 : \tau}{\Gamma \vdash e_1 \leftarrow e_2 : \mathtt{U}} \qquad \frac{\Gamma \vdash e_1 : \mathtt{U} \qquad \Gamma \vdash e_2 : \tau}{\Gamma \vdash e_1; e_2 : \tau}$$