POSTECH PL 연구분야 소개

박성우

ROSAEC Kick-off Workshop

2008년 11월 22일

미래의 프로그래밍 언어

1. 함수형 프로그래밍
   - 생산성↑, 속도↑, 사용계층↑
   - 응용 영역이 계속 확장 중
2. 병렬 프로그래밍
   - 멀티코어/Manycore 환경을 위해서는 필수
3. Computational effect 추적
   - Haskell과 ML을 구별하는 경계
   - Call-by-name + call-by-value가 필요

현재 연구 주제

- 멀티코어 환경을 위한 함수형 언어
- λ-Calculus에 기초한 함수형 하드웨어기술 언어
- Modal logic and proof theory
- 멀티코어 환경에서 병렬 계산

멀티코어 환경을 위한 함수형 언어

멀티코어 환경을 대상으로하는 함수형
병렬 프로그래밍 언어를 설계하고
구현한다.

대세는 멀티코어

- 코어 수의 증가

source: Herb Sutter - "Software and the concurrency revolution"

- Intel, 80-cores, 2011

Effect 추적

- Call-by-value 함수형 언어 가정
- 연산식 계산 결과가 pure인지 아닌지 추적
  - *box M : ⊡ A*

    *M*의 계산 결과는 pure (포인터가 없음)
- 응용
  - Task-parallelism에서 메시지 전송의 안전성
  - Data-parallelism에서 data race 방지
    - *mapP f* $[e_1, e_2, \ldots, e_n]$

Base 언어에 독립적인 모듈 시스템

- ML 모듈 시스템
  - functor + module
  - SML: Base 언어에 종속
  - OCAML: Better, but still not general
- 목표
  - Base 언어에 완전히 독립적인 모듈 시스템 설계
  - 모듈 시스템이 없는 언어에 functor를 제공
- 현재 safety 증명 중

Lax Logic 언어

- Lax logic
  - Computational effect의 논리적 특성
    - If $A\ true$, then $A\ lax$.
    - If $\Gamma \vdash A\ lax$ and $\Gamma, A\ true \vdash C\ lax$, then $\Gamma \vdash C\ lax$.
- Lax logic에서 시작하는 base 언어 구현
  - Pure 계산과 impure 계산을 구별하는 언어가 유도됨

    $\Rightarrow$ Call-by-name + call-by-value
  - One-pass CPS transformation이 유도됨

파서 생성기 생성기

- Compiler를 자체적으로 self-compile하기 위해서는 파서 생성기가 필요
- OCAML 파서 생성기는 OCAML에 종속
- 목표
  - 모든 함수형 언어에서 이용할 수 있는 파서 생성기 생성기를 설계
  - 대상 언어의 기본 문법 구조를 지정하면 Byacc에 기초한 파서 생성기를 생성함
    - 함수, 패턴검사, ...
- 구현 완료, 시험 중 (Haskell, Ocaml, SML)

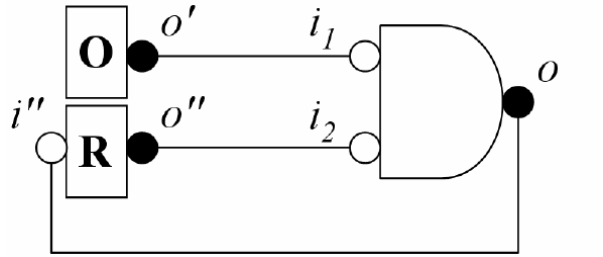
λ-Calculus에 기초한 함수형 하드웨어기술 언어

λ–Calculus에서 시작하는
하드웨어기술 언어를 설계하고
구현한다.

Core Language lλ

- 기존형 함수형 하드웨어기술 언어
  - 주로 Haskell 등에 embedded 되어 있음
- 목표
  - λ-Calculus를 하드웨어기술 언어로 이용
- 언어 정의 및 구현 완료
  - l = linear (하드웨어를 여러번 쓸 수 없음)

$$\mathbf{fix}\ x : \mathbf{1}.\ \mathtt{and}\ \mathtt{zero}\ (\mathtt{reg}\ x)$$



Fourier 변환

$$
\begin{array}{lcll}
\texttt{twoC} & : & \forall\alpha.(\alpha\to\alpha)\multimap^2(\alpha^2\to\alpha^2) & = \quad \Lambda\alpha.\hat{\lambda}f^2{:}\alpha\to\alpha.\,\lambda(x,y){:}\alpha^2.\,(f\,x, f\,y)\\
\texttt{prodC} & : & \forall\alpha.(\alpha^2\to\alpha)\multimap^2(\alpha^4\to\alpha^2) & = \quad \Lambda\alpha.\hat{\lambda}f^2{:}\alpha^2\to\alpha.\,\lambda((x,y),(z,w)){:}\alpha^4.\,(f\,(x,z), f\,(y,w))\\
\texttt{riffleC} & : & \forall\alpha.(\alpha^2\to\alpha^2)\multimap^2(\alpha^4\to\alpha^4) & = \quad \Lambda\alpha.\hat{\lambda}f^2{:}\alpha^2\to\alpha^2.\,\lambda((x,y),(z,w)){:}\alpha^4.\,(f\,(x,z), f\,(y,w))\\
\texttt{unriffleC} & : & \forall\alpha.(\alpha^2\to\alpha^2)\multimap^2(\alpha^4\to\alpha^4) & = \quad \Lambda\alpha.\hat{\lambda}f^2{:}\alpha^2\to\alpha^2.\,\lambda(p,q){:}\alpha^4.\,\mathbf{proj}\;f\;p\;\mathbf{of}\;(x,z)\;\mathbf{in}\;\mathbf{proj}\;f\;q\;\mathbf{of}\;(y,w)\;\mathbf{in}\;((x,y),(z,w))
\end{array}
$$

$$
\begin{array}{lcll}
\texttt{riffle}_1 & : & \texttt{c}^2\to\texttt{c}^2 & = \quad \lambda p{:}\texttt{c}^2.\,p\\
\texttt{riffle}_2 & : & \texttt{c}^4\to\texttt{c}^4 & = \quad (\texttt{riffleC}\,\langle\texttt{c}\rangle)\hat{}^2\texttt{riffle}_1\\
\texttt{riffle}_3 & : & \texttt{c}^8\to\texttt{c}^8 & = \quad (\texttt{riffleC}\,\langle\texttt{c}^2\rangle)\hat{}^2\texttt{riffle}_2\\
\texttt{riffle}_4 & : & \texttt{c}^{16}\to\texttt{c}^{16} & = \quad (\texttt{riffleC}\,\langle\texttt{c}^4\rangle)\hat{}^2\texttt{riffle}_3
\end{array}
$$

$$
\begin{array}{lcll}
\texttt{unriffle}_1 & : & \texttt{c}^2\to\texttt{c}^2 & = \quad \lambda p{:}\texttt{c}^2.\,p\\
\texttt{unriffle}_2 & : & \texttt{c}^4\to\texttt{c}^4 & = \quad (\texttt{unriffleC}\,\langle\texttt{c}\rangle)\hat{}^2\texttt{unriffle}_1\\
\texttt{unriffle}_3 & : & \texttt{c}^8\to\texttt{c}^8 & = \quad (\texttt{unriffleC}\,\langle\texttt{c}^2\rangle)\hat{}^2\texttt{unriffle}_2\\
\texttt{unriffle}_4 & : & \texttt{c}^{16}\to\texttt{c}^{16} & = \quad (\texttt{unriffleC}\,\langle\texttt{c}^4\rangle)\hat{}^2\texttt{unriffle}_3
\end{array}
$$

$$
\begin{array}{lcll}
\texttt{g}_1 & : & \texttt{c}^2\to\texttt{c}^2 & = \quad \lambda p{:}\texttt{c}^2.\,(\texttt{cplus}\;p, \texttt{cminus}\;p)\\
\texttt{g}_2 & : & \texttt{c}^4\to\texttt{c}^4 & = \quad \texttt{twoC}\,\langle\texttt{c}^2\rangle\hat{}^2\texttt{g}_1\\
\texttt{g}_3 & : & \texttt{c}^8\to\texttt{c}^8 & = \quad \texttt{twoC}\,\langle\texttt{c}^4\rangle\hat{}^2\texttt{g}_2\\
\texttt{g}_4 & : & \texttt{c}^{16}\to\texttt{c}^{16} & = \quad \texttt{twoC}\,\langle\texttt{c}^8\rangle\hat{}^2\texttt{g}_3
\end{array}
$$

$$
\begin{array}{lcll}
\texttt{bfly}_1 & : & \texttt{c}^2\to\texttt{c}^2 & = \quad \texttt{unriffle}_1\circ\texttt{g}_1\circ\texttt{riffle}_1\\
\texttt{bfly}_2 & : & \texttt{c}^4\to\texttt{c}^4 & = \quad \texttt{unriffle}_2\circ\texttt{g}_2\circ\texttt{riffle}_2\\
\texttt{bfly}_3 & : & \texttt{c}^8\to\texttt{c}^8 & = \quad \texttt{unriffle}_3\circ\texttt{g}_3\circ\texttt{riffle}_3\\
\texttt{bfly}_4 & : & \texttt{c}^{16}\to\texttt{c}^{16} & = \quad \texttt{unriffle}_4\circ\texttt{g}_4\circ\texttt{riffle}_4
\end{array}
$$

$$
\begin{array}{lcll}
\texttt{prod}_1 & : & \texttt{c}^2\to\texttt{c} & = \quad \texttt{cmult}\\
\texttt{prod}_2 & : & \texttt{c}^4\to\texttt{c}^2 & = \quad \texttt{prodC}\,\langle\texttt{c}\rangle\hat{}^2\texttt{prod}_1\\
\texttt{prod}_3 & : & \texttt{c}^8\to\texttt{c}^4 & = \quad \texttt{prodC}\,\langle\texttt{c}^2\rangle\hat{}^2\texttt{prod}_2\\
\texttt{prod}_4 & : & \texttt{c}^{16}\to\texttt{c}^8 & = \quad \texttt{prodC}\,\langle\texttt{c}^4\rangle\hat{}^2\texttt{prod}_3
\end{array}
$$

$$
\begin{array}{lcll}
\texttt{factor}_1 & : & \texttt{c} & = \quad W_2^0\\
\texttt{factor}_2 & : & \texttt{c}^2 & = \quad (W_4^0, W_4^1)\\
\texttt{factor}_3 & : & \texttt{c}^4 & = \quad ((W_8^0, W_8^1), (W_8^2, W_8^3))\\
\texttt{factor}_4 & : & \texttt{c}^8 & = \quad (((W_{16}^0, W_{16}^1), (W_{16}^2, W_{16}^3)), ((W_{16}^4, W_{16}^5), (W_{16}^6, W_{16}^7)))
\end{array}
$$

$$
\begin{array}{lcll}
\texttt{f}_1 & : & \texttt{c}^2\to\texttt{c}^2 & = \quad \lambda(x,y){:}\texttt{c}^2.\,\texttt{bfly}_1\,(x, \texttt{prod}_1\,(y, \texttt{factor}_1))\\
\texttt{f}_2 & : & \texttt{c}^4\to\texttt{c}^4 & = \quad \lambda(x,y){:}\texttt{c}^4.\,\texttt{bfly}_2\,(x, \texttt{prod}_2\,(y, \texttt{factor}_2))\\
\texttt{f}_3 & : & \texttt{c}^8\to\texttt{c}^8 & = \quad \lambda(x,y){:}\texttt{c}^8.\,\texttt{bfly}_3\,(x, \texttt{prod}_3\,(y, \texttt{factor}_3))\\
\texttt{f}_4 & : & \texttt{c}^{16}\to\texttt{c}^{16} & = \quad \lambda(x,y){:}\texttt{c}^{16}.\,\texttt{bfly}_4\,(x, \texttt{prod}_4\,(y, \texttt{factor}_4))
\end{array}
$$

$$
\begin{array}{lcll}
\texttt{fft}_1 & : & \texttt{c}^2\to\texttt{c}^2 & = \quad \texttt{f}_1\\
\texttt{fft}_2 & : & \texttt{c}^4\to\texttt{c}^4 & = \quad \texttt{f}_2\circ(\texttt{twoC}\,\langle\texttt{c}^2\rangle\,\texttt{f}_1)\\
\texttt{fft}_3 & : & \texttt{c}^8\to\texttt{c}^8 & = \quad \texttt{f}_3\circ(\texttt{twoC}\,\langle\texttt{c}^4\rangle\,\texttt{f}_2)\\
\texttt{fft}_4 & : & \texttt{c}^{16}\to\texttt{c}^{16} & = \quad \texttt{f}_4\circ(\texttt{twoC}\,\langle\texttt{c}^8\rangle\,\texttt{f}_3)
\end{array}
$$

Iλ의 확장

- Polymorphism
  - 구조는 같고 wire의 폭만 다른 회로를 하나로 표현할 수 있다
- ML 스타일 모듈 시스템 (functor, module)
  - 하드웨어 설계의 재사용성 증가
- Type analysis
  - 타입에 따른 프로그램 생성
  - $add : 1^n \to 1^n \to 1^n \times 1$
- Cf. HFL Workshop, ETAPS 2009

Modal Logic and Proof Theory

기초 연구를 계속한다.

Modal Logic Internalizing Normal Proofs

- 새로운 modality △

$$\frac{A\uparrow}{\triangle A\ true}\ \triangle\mathsf{I}$$

- Sequent calculus 완성, Cut-elimination 증명

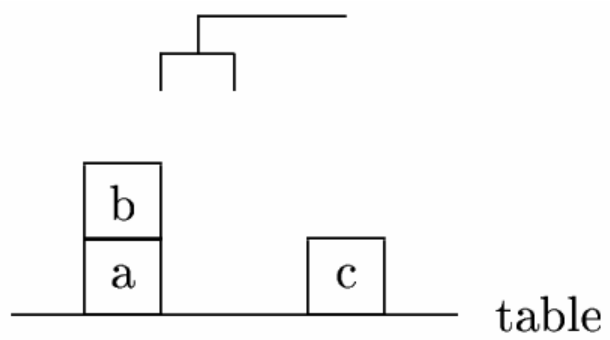
- Coq으로 proof를 mechanize
- Twelf로 proof를 mechanize (예정)

Theorem Prover for BI

- BI
  - Logic of Bunched Implications
  - Separation logic과 밀접한 관계
    - Separation logic이 BI 모델의 일종
- 기존의 theorem prover for BI
  - BILL
  - Inverse method prover
- 목표
  - Inverse method + focusing

Linear Logic for Database Theory

- Planning and frame problem



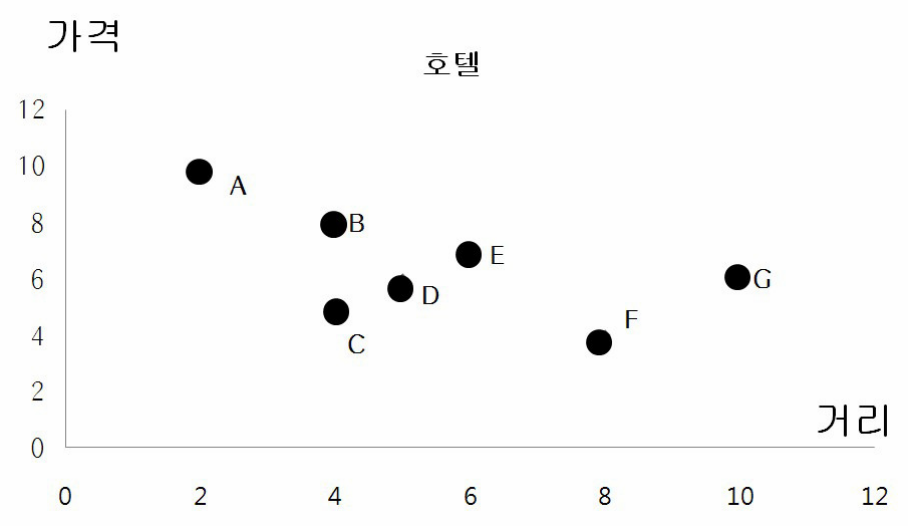
  - Situation calculus: 일차논리에 기초, 복잡
  - Linear logic: 간결하고 elegant한 해결책 제공
- 데이터베이스 이론에서 frame problem이 자주 등장
  - Linear logic 이용한 연구는 거의 없음
  - Eg. Linear logic for the semantics of nondeterministic databases [ICDT 2009]

멀티코어 환경에서 병렬 계산

실제 병렬 프로그래밍을 해 본다.

Parallel Skyline

- Skyline 계산의 병렬화



- 기존의 알고리즘 (BBS) 병렬화
- 새로운 병렬 알고리즘 설계
- 새로운 순차 알고리즘 설계 시험 중

Parallel DOM

- DOM (Document Object Model)
  - HTML Browser의 핵심 엔진
  - 객체지향 방식으로 설계된 tree 구조
  - 현재 모든 브라우저는 순차적 계산을 수행
- Parallel DOM
  - 여러개의 DOM 연산을 동시에 허용
  - Haskell의 transactional memory를 이용해서 구현
- Transaction memory = 병렬 프로그래밍의 구원자?

ROSAEC 연구

Wild Idea

- 논리 기반 순차적 프로그램 분석
  - Proof of the presence of bugs
  - Proof를 찾으면 counter-example 생성
  - Proof를 못찾으면 신뢰성 증가
  - Programmer의 proof annotation이 proof 찾을 가능성을 높임
  - ???

- 병렬 프로그램의 분석으로 확장

감사합니다