확률적 프로그램의 샘플링 수를 줄이는 최적화 과정 디자인

김윤승
서울대학교 소프트웨어 원리 연구실

@ ROSAEC Workshop

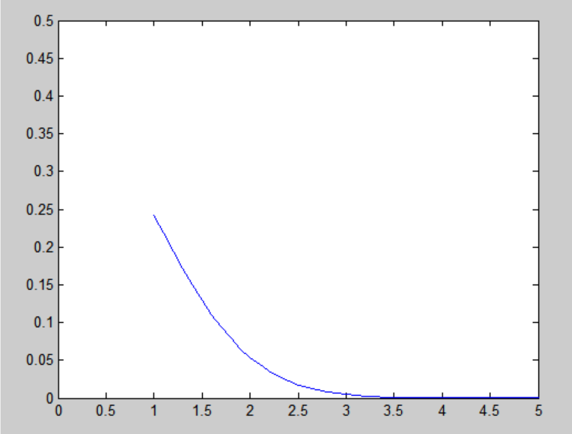
2015. 1. 27.

확률적 프로그래밍과 우리의 연구

- 사용자가 확률 문제를 프로그램 형식으로 정의한다
- “확률 추론 엔진“ 프로그램이, 정의된 “확률적 프로그램”을 읽고 문제를 푼다

```
x ~ Gaussian(0,1);
observe (x > 1);
return x
```

추론 엔진



연구 목적:
1. 확률적 프로그램을 잘 변환하면 추론엔진이 더 쉽고 빠르게 문제를 풀 수 있을 것이다. 그런 변환을 찾아내서 정의하자.
2. 그런 변환을 하는 최적화 과정을 구현해서 현재 Microsoft Research에서 개발중인 추론 엔진 R2(http://research.microsoft.com/en-us/projects/r2/)의 성능을 높이자

샘플링 수를 줄이기

- 똑같은 확률분포를 나타내는 두 프로그램이 있다면,
- 대체로 샘플링 수가 적을 수록 “추론 엔진”이 분포를 알아내기 쉽다.

```
m ~ Gaussian(0,1);
x ~ Gaussian(m,1);
y ~ Gaussian(m,1);
observe (x + y > 1);

return m
```

```
m ~ Gaussian(0,1);
z ~ Gaussian(2m,sqrt(2));

observe (z > 1);

return m
```

프로그램 변환 종류

- Linear Gaussians 합치기

```
x ~ Gaussian(m1,v1)
y ~ Gaussian(m2,v2)
..
j = px+qy
..
observe (px+qy < 1)
---------------------------------
z ~ Gaussian(pm1+pm2,..)
..
j = z
..
observe (z < 1)
```

- Sampling statement 지우기 (연속 분포)

```
..
x ~ Gaussian(m,v)
observe (a < x < b)
..
---------------------------------
weight (PDFGaus(m,v,b) –
          PDFGaus(m,v,a))
...
```

- Sampling statement 지우기 (이산 분포)

```
..
x ~ flip (0.7)
observe ( x V y )
..
---------------------------------
..
weight ( 0.7 + 0.3*X(y) )
..
```

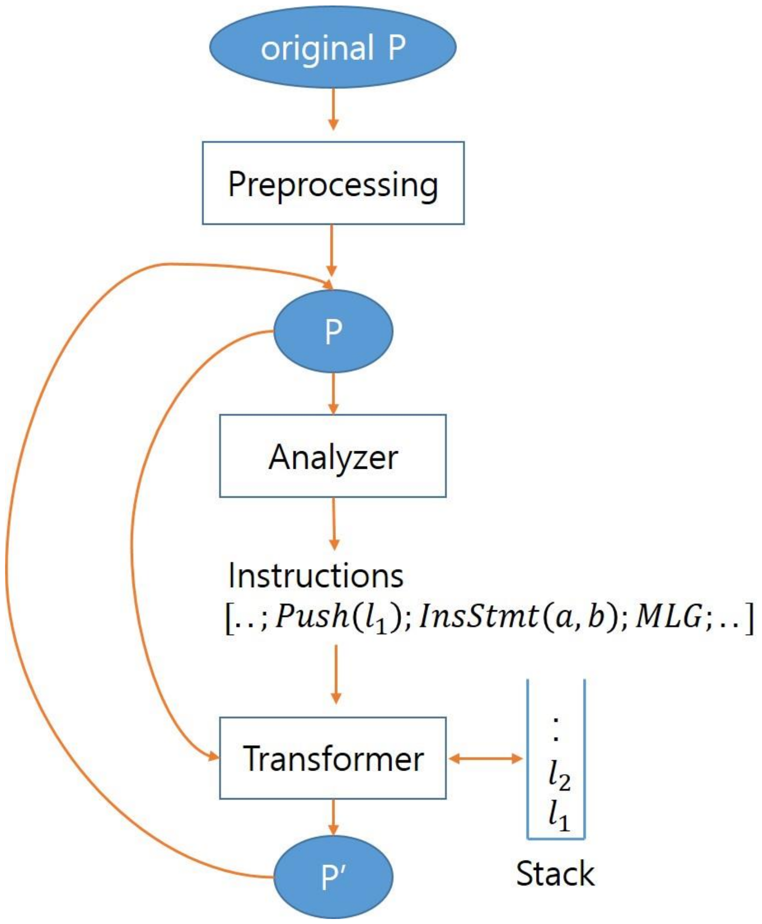
풀어야 할 난관

- 프로그램을 읽고, 최적화 변환이 가능한 부분을 최대한 많이 찾아내야 한다.
- 변환 전·후의 프로그램이 프로그램의 의미(확률 분포)를 변화시키는 일이 없어야 한다.

우리의 전략

- 최적화 과정을 분석기와 변환기로 분리한다
  - 분석 과정: 코드를 스캔해서 어느 위치에 어떤 변환을 적용할 지 변환기에 알려줌
  - 변환 과정: 하나의 최적화를 연쇄적인 작은 프로그램 변환으로 분리
    - 변환이 옳다는 것을 증명하기 쉬움
    - 분석기가 알려준 정보가 허위였을 경우, fail
    - 핵심적인 파트를 한 군데로 모아서, 부분을 변환하고 컨텍스트에 바꿔끼우기

전체 최적화 프로세스



- Preprocessing
- Analyzer

  수행할 변환의 종류, 순서, 적용 위치 정보를 생성
- Transformer

  Analyzer가 내놓은 변환 과정을 순서대로 수행

Analyzer의 결과와 Stack의 정보에 관계없이, Transformer에서 수행하는 변환이 프로그램의 의미를 바꾸지 않음이 증명되면 이 최적화기는 믿을 수 있다.

(시간이 남으면..) 증명 원리

- 확률적 프로그램 코드의 일부를 바꿔 끼웠을 때,
  - C[P] → C[P']

C와 P, P'이 특정한 관계를 만족시키면,

[[C[P]]] = [[C[P']]] 임을 증명하는 방법을 고안함

증명에 사용할 Relation

$$\frac{\quad V_1 \quad}{\dfrac{P \qquad P'}{V_2}}$$

$$\frac{\{x, y\}}{\dfrac{\begin{array}{ll} x{\sim}Gaus(0,1) & x{\sim}Gaus(0,1) \\ z{\sim}Gaus(y,1) & z{\sim}Gaus(y,1) \end{array}}{\{y, z\}}}$$

- V_1, V_2 : set of variables
- P, P' : programs

- 두 프로그램을 실행하기 전의 메모리 상태가 최대 V_1를 제외하고 같다면,
- 두 프로그램을 실행한 후 변수값의 분포는 V_2를 제외하고 같음

증명할 목표

원래 프로그램이 P return E 일 때,

$$\cfrac{\cfrac{\emptyset}{P \qquad P'}}{V} \qquad \land \qquad V \mathbin{\#} var(E)$$

General Rules

$$\frac{V}{\text{x =E} \quad \text{x=E}}\ \text{if } var(E)\#V$$
$$V - \{x\}$$

$$\dfrac{\dfrac{V_1}{\dfrac{\text{P1} \quad \text{P1}'}{V_2}} \qquad \dfrac{V_2}{\dfrac{\text{P2} \quad \text{P2}'}{V_3}}}{\dfrac{V_1}{\dfrac{\text{P1 ; P2} \quad \text{P1}' \text{ ; P2}'}{V_3}}}$$

$$\dfrac{\dfrac{V_1}{\dfrac{\text{P1} \quad \text{P1}'}{V_2}} \qquad \dfrac{V_1}{\dfrac{\text{P2} \quad \text{P2}'}{V_2}} \qquad E\#V_1}{\dfrac{V_1}{\dfrac{\text{if E P1 else P2} \quad \text{if E P1}' \text{ else P2}'}{V_2}}}$$

부분 변환에 대한 증명

- MergeLinearGaussians
- SamplingElimination_cont
- SamplingElimination_disc

$$\{x, y, ..\}$$

x ~ Gaussian(a,b)
y ~ Gaussian(c,d)
z = px + qy

z = Gaussian (pa+qc,
sqrt((pb)^2 + (qd)^2))

$$\{x, y\}$$

$$\{x, ..\}$$

x ~ Gaussian(m,v)
observe (a < x < b)

weight (PDFGaus(m,v,b)
– PDFGaus(m,v,a))

$$\{x\}$$

예제

		$\emptyset$
`m ~ Gaussian(0,1)`	`m ~ Gaussian(0,1)`	
		$\emptyset$

`x ~ Gaussian(a,1)`
`y ~ Gaussian(a,1)`
`z = x + y`
	`z ~ Gaussian(2a, sqrt(2))`	

		$\{x, y\}$
`observe (z > 1)`	`observe (z > 1)`	
		$\{x, y\}$
`return m`	`return m`	

감사합니다