

한국정보과학회
프로그래밍언어연구회
겨울학교

SIGPL Winter School 2023

2023년 2월 22~24일
고려대학교

회상

+

음악프로그래밍

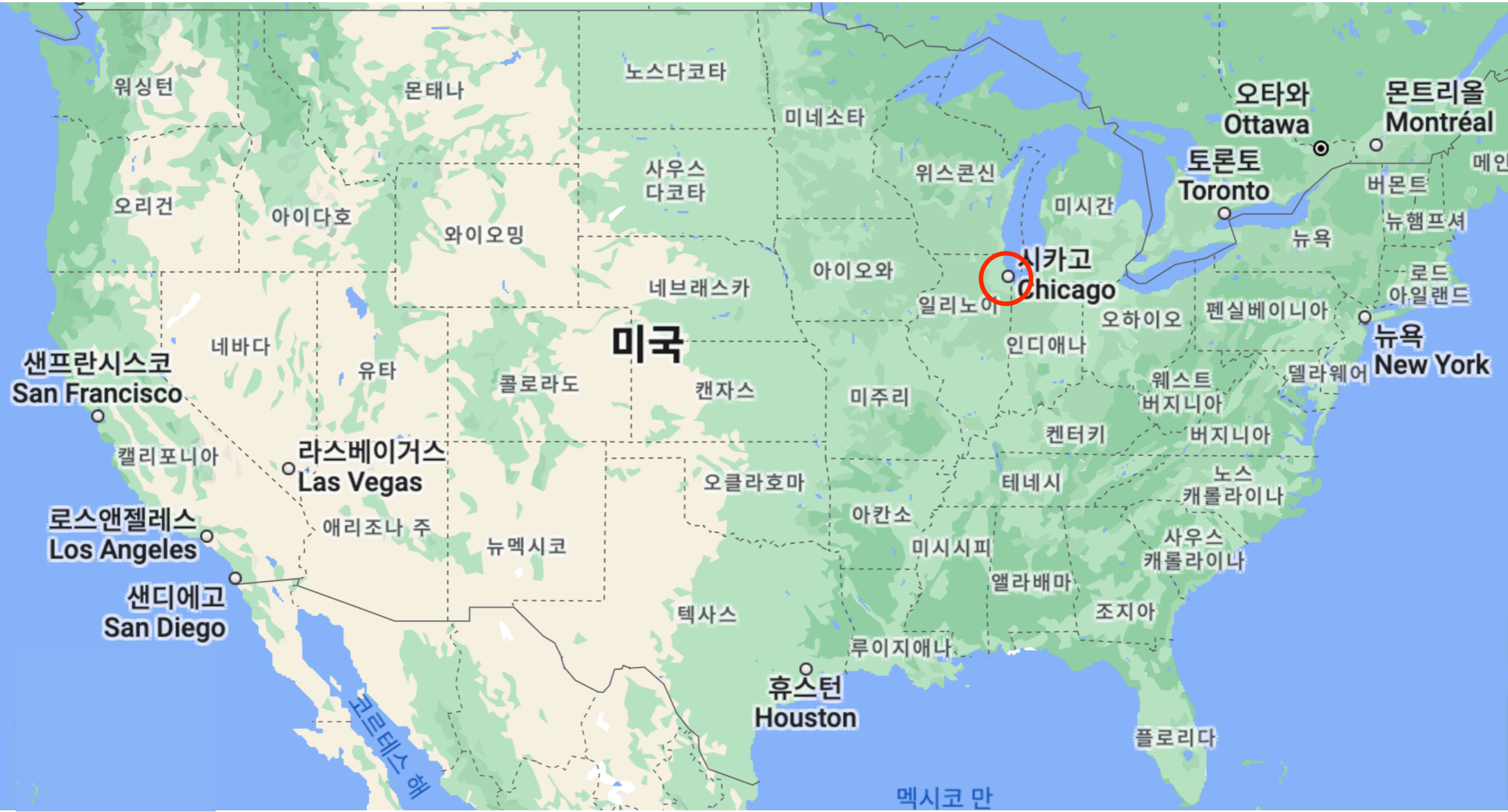
한양대학교 ERICA
도경구

IE

=

>

CS



워싱턴

몬태나

노스다코타

미네소타

오타와
Ottawa

몬트리올
Montréal

오리건

아이다호

와이오밍

사우스
다코타

위스콘신

미시간

토론토
Toronto

뉴욕

버몬트

뉴햄프셔

아이다호

와이오밍

네브래스카

아이오와

일리노이

시카고
Chicago

오하이오

펜실베이니아

로드
아일랜드

뉴욕
New York

샌프란시스코
San Francisco

네바다

유타

콜로라도

미국

캔자스

미주리

인디애나

웨스트
버지니아

델라웨어

캘리포니아

라스베이거스
Las Vegas

콜로라도

캔자스

미주리

켄터키

버지니아

델라웨어

뉴욕
New York

로스앤젤레스
Los Angeles

애리조나 주

뉴멕시코

오클라호마

아칸소

테네시

노스
캐롤라이나

샌디에고
San Diego

애리조나 주

뉴멕시코

텍사스

아칸소

미시시피

앨라배마

사우스
캐롤라이나

조지아

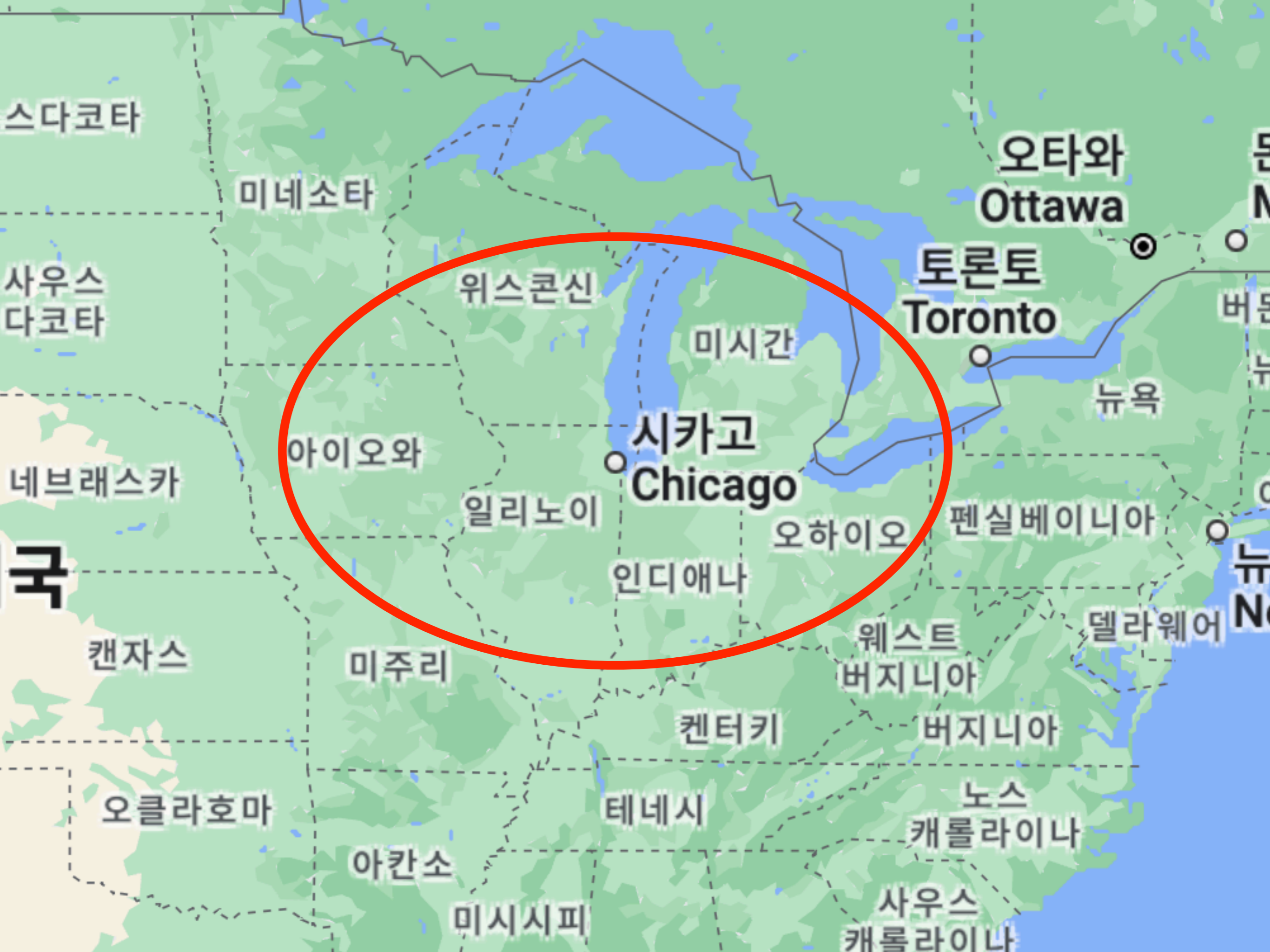
휴스턴
Houston

루이지애나

플로리다

카리브해

멕시코 만



스다코타

미네소타

오타와
Ottawa

사우스
다코타

위스콘신

미시간

토론토
Toronto

뉴욕

네브래스카

아이오와

시카고
Chicago

일리노이

오하이오

펜실베이니아

국

인디애나

캔자스

미주리

웨스트
버지니아

델라웨어

켄터키

버지니아

오클라호마

테네시

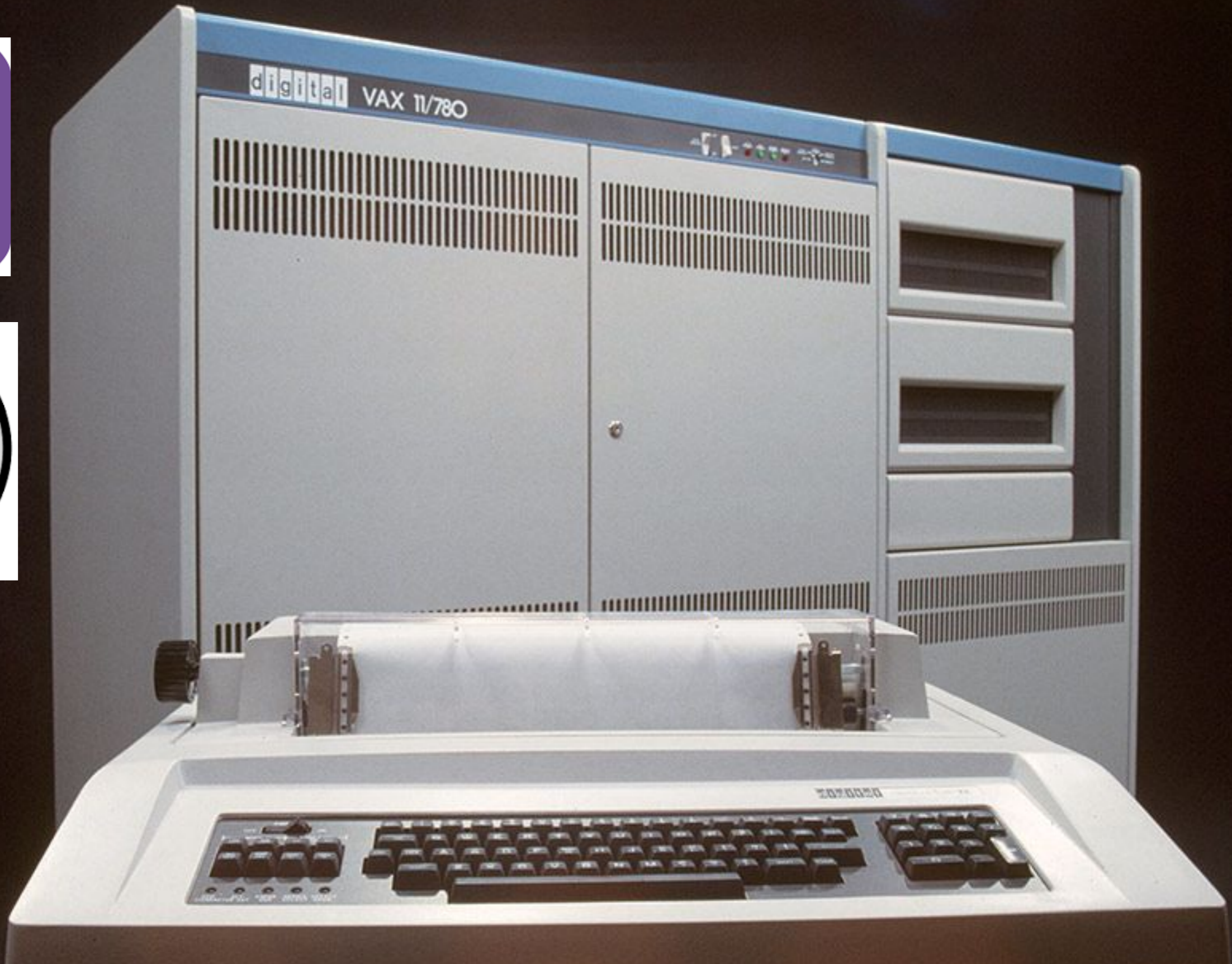
노스
캐롤라이나

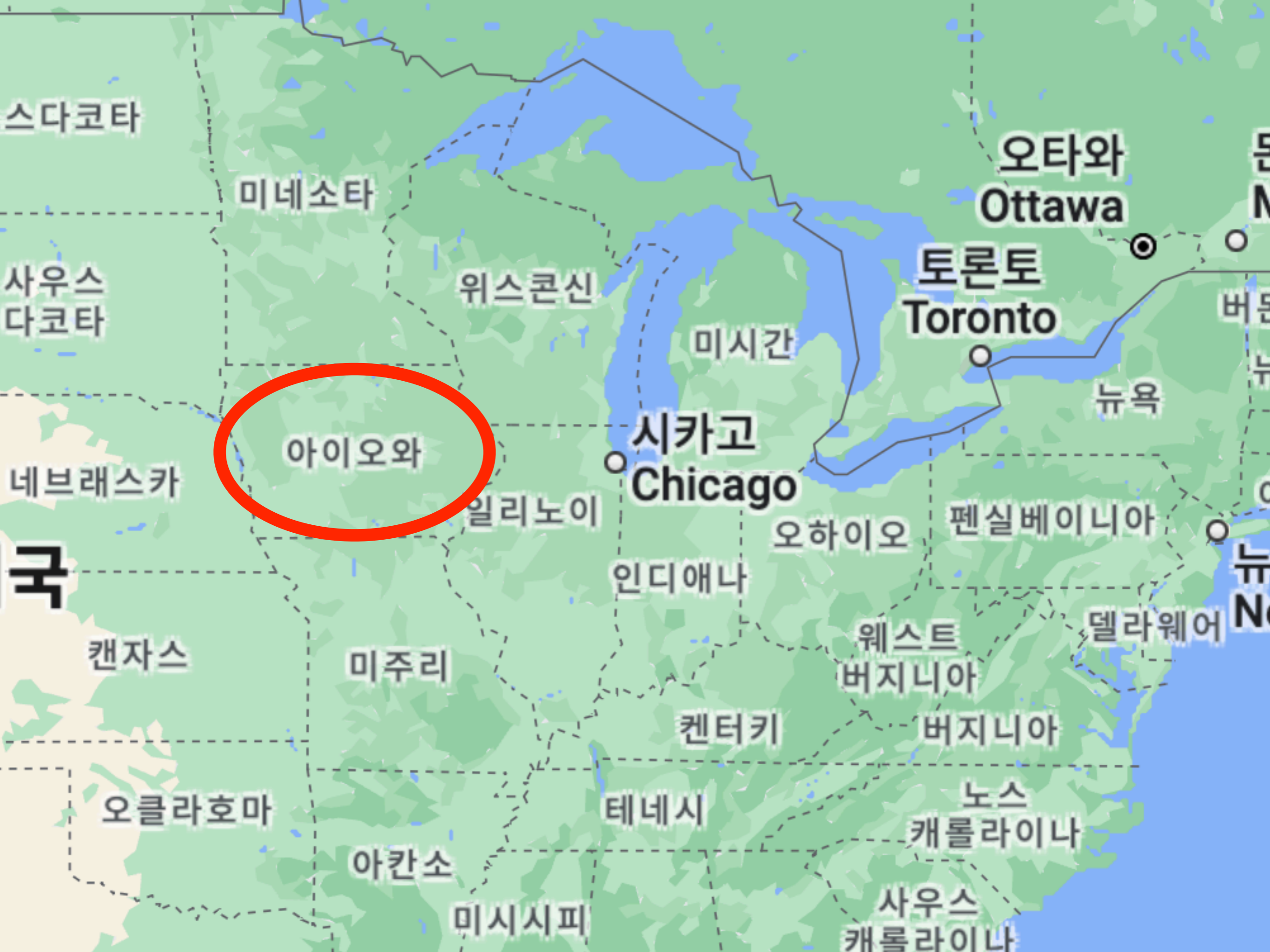
아칸소

미시시피

사우스
캐롤라이나

digital VAX 11/780





스다코타

미네소타

오타와
Ottawa

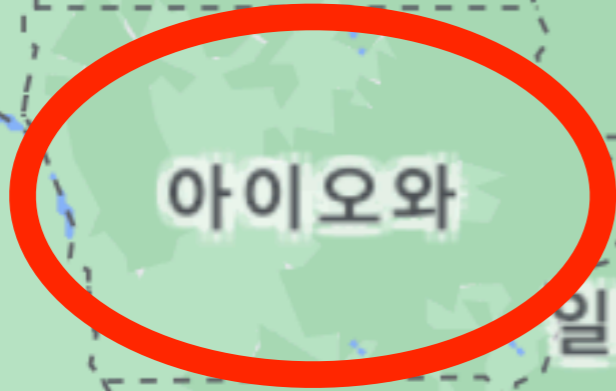
사우스
다코타

위스콘신

토론토
Toronto

미시간

뉴욕



아이오와

시카고
Chicago

네브래스카

일리노이

오하이오

펜실베이니아

국

인디애나

캔자스

미주리

웨스트
버지니아

델라웨어

켄터키

버지니아

오클라호마

테네시

노스
캐롤라이나

아칸소

미시시피

사우스
캐롤라이나

1984년 6월

IOWA STATE UNIVERSITY
Department of Computer Science



John Atanasoff



PASCAL



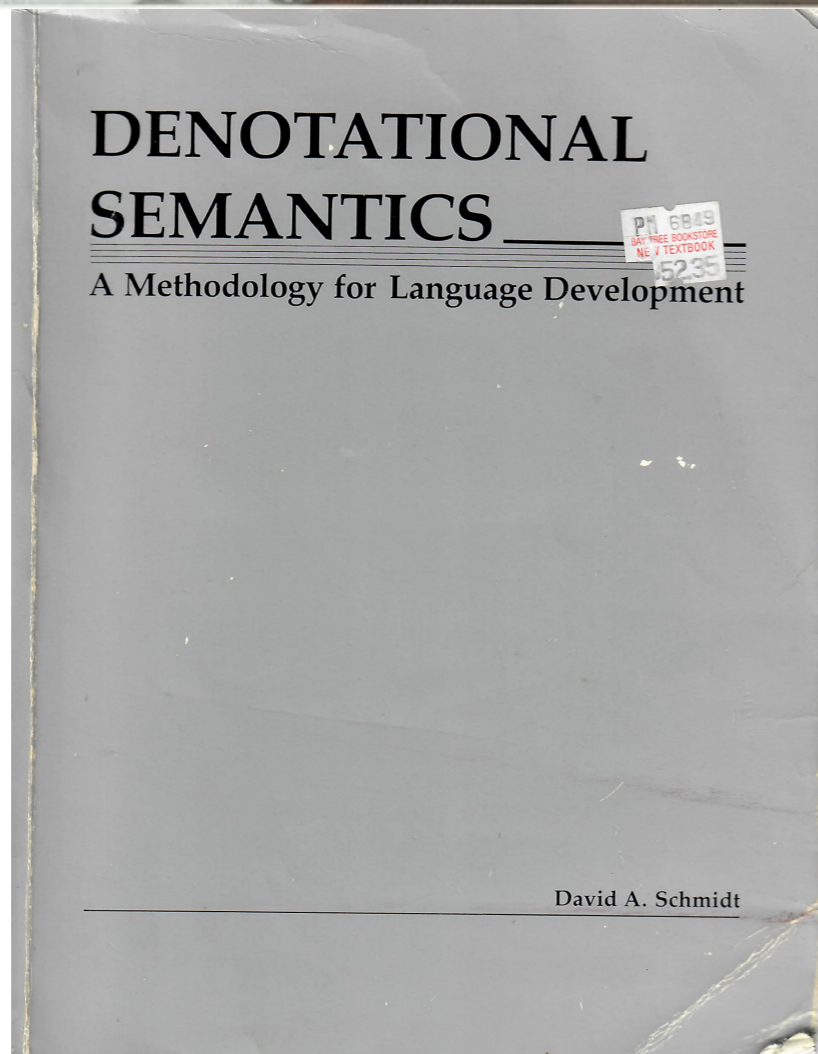
CS

>

PL



David A. Schmidt





**ML Interpreter by
Luca Cardelli**



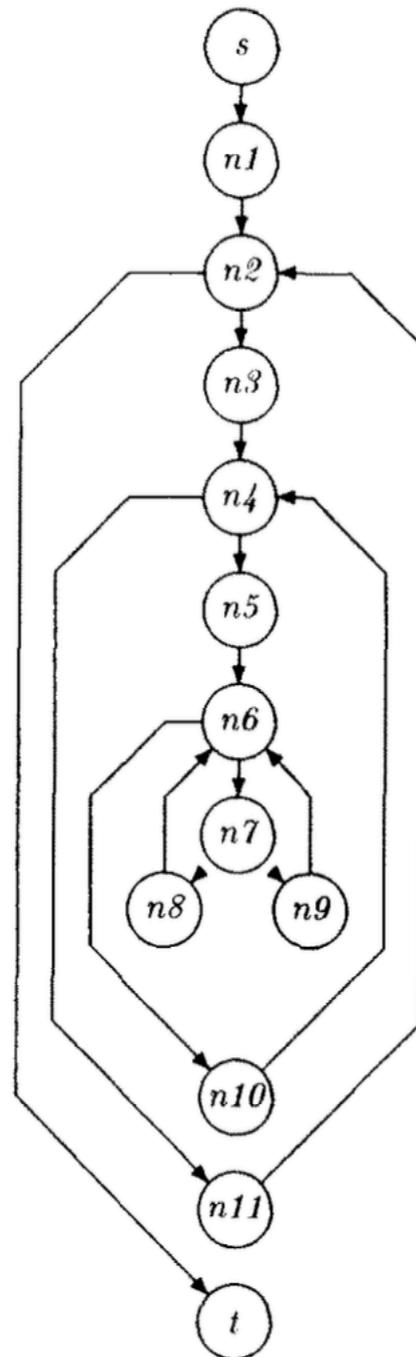
Master's Degree Report

Generating A Standard Representation From Pascal Programs

```

procedure Shellsort (var A: array[1..n] of integer );
var
    i, j, incr: integer;
begin
    incr := n div 2;
    while incr > 0 do begin
        for i := incr + 1 to n do begin
            j := i - incr;
            while j > 0 do
                if A[j] > A[j + incr] then begin
                    swap(A[j], A[j + incr]);
                    j := j - incr;
                end
                else
                    j := 0 (* break *)
                end;
            incr := incr div 2;
        end
    end (* Shellsort *) ;

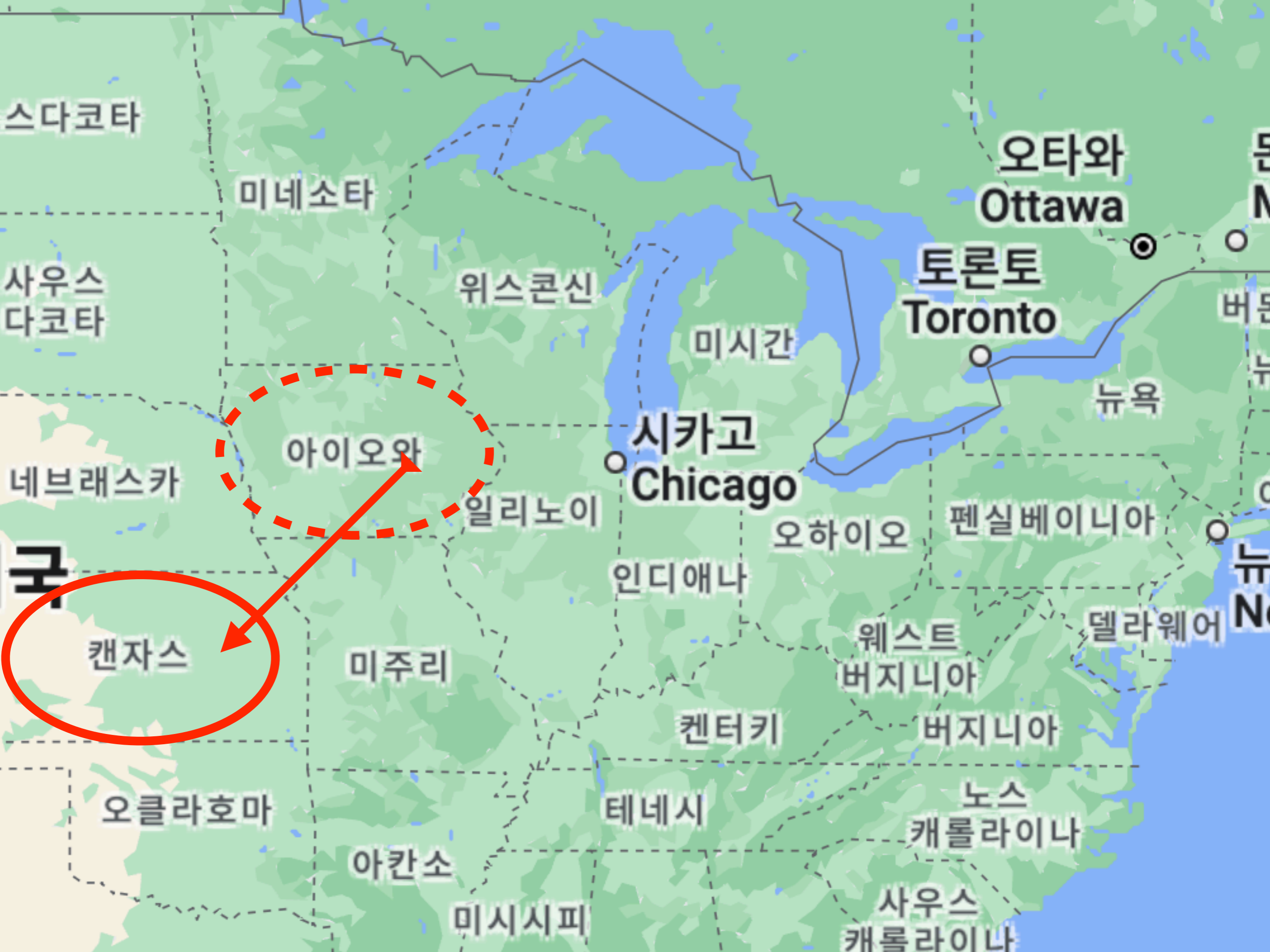
```



```

((Shellsort,⟨A⟩,⟨n⟩),
({(s,⟨⟩,⟨⟩,⟨{⟩,⟨{⟩},
(n1,⟨(incr,⟨n,2⟩),⟨⟩,
  (⟨:=,1⟩,⟨div,1⟩,⟨;,1⟩,⟨begin...end,1⟩), {⟨n,1⟩,⟨incr,1⟩,⟨1,1⟩} } ),
(n2,⟨⟩,⟨incr,0⟩, (⟨while...do,1⟩,⟨>,1⟩), {⟨incr,1⟩,⟨1,1⟩} } ),
(n3,⟨(i,⟨incr,1⟩),⟨⟩,
  (⟨+,1⟩,⟨begin...end,1⟩, ⟨for...to...do,1⟩), {⟨incr,1⟩,⟨i,1⟩,⟨1,1⟩} } )
(n4,⟨⟩,⟨i,n⟩, (⟨{⟩,⟨{n,1⟩} } ),
(n5,⟨(j,⟨i,incr⟩),⟨⟩,
  (⟨:=,1⟩,⟨-,1⟩,⟨;,1⟩,⟨begin...end,1⟩), {⟨i,1⟩,⟨incr,1⟩,⟨j,1⟩} } ),
(n6,⟨⟩,⟨j,0⟩, (⟨while...do,1⟩,⟨>,1⟩), {⟨j,1⟩} } ),
(n7,⟨⟩,⟨A,j,A,j,incr⟩,
  (⟨if...then,1⟩,⟨[],2⟩,⟨+,1⟩,⟨>,1⟩), {⟨j,2⟩,⟨a,2⟩,⟨incr,1⟩} } ),
(n8,⟨(swap,⟨(A,j), ⟨A,j,incr⟩),⟨j,⟨j,incr⟩⟩), ⟨⟩,
  (⟨swap,1⟩,⟨(),1⟩,⟨[],2⟩,⟨;,1⟩,⟨+,1⟩, ⟨;,1⟩,⟨:=,1⟩,⟨-,1⟩,⟨begin...end,1⟩),
  {⟨j,2⟩,⟨a,2⟩,⟨incr,1⟩} } ),
(n9,⟨(j,⟨0⟩),⟨⟩,⟨(else,1), (:=,1)⟩, {⟨j,1⟩,⟨0,1⟩} } ),
(n10,⟨(i,⟨i⟩),⟨⟩, (⟨{⟩,⟨{⟩} } ),
(n11,⟨(incr,⟨incr,2⟩),⟨⟩, (⟨:=,1⟩,⟨div,1⟩,⟨;,1⟩), {⟨incr,2⟩,⟨2,1⟩} } ),
(t,⟨⟩,⟨⟩,⟨{⟩,⟨{⟩} } )
{ (s,n1), (n1,n2), (n2,n3), (n2,t), (n3,n4),
  (n4,n5), (n4,n11), (n5,n6), (n6,n7),
  (n6,n10), (n7,n8), (n7,n9), (n8,n6),
  (n9,n6), (n10,n4), (n11,n2) },
s,
t))

```



국

캔자스

아이오와

시카고
Chicago

토론토
Toronto

오타와
Ottawa

위스콘신

미시간

미네소타

스다코타

사우스
다코타

네브래스카

일리노이

오하이오

펜실베이니아

인디애나

웨스트
버지니아

델라웨어

미주리

켄터키

버지니아

오클라호마

테네시

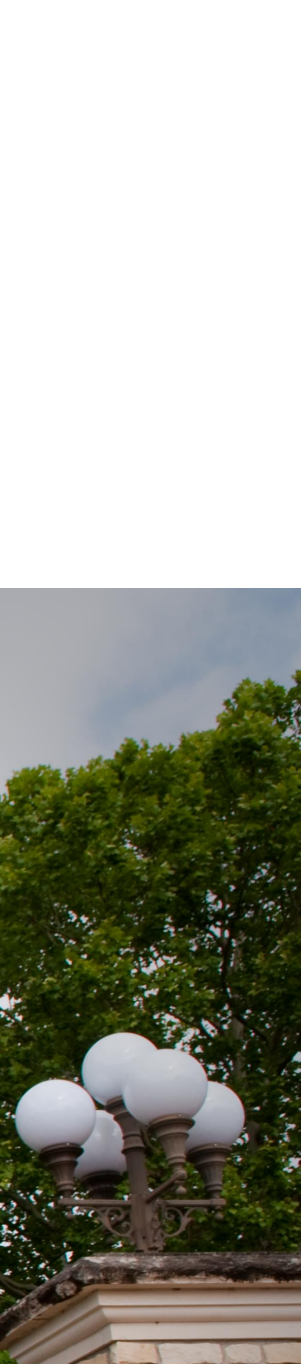
노스
캐롤라이나

아칸소

미시시피

사우스
캐롤라이나





1987년 1월



Olivier Danvy



David A. Schmidt



George Strecker



Partial Evaluation

Denotational Semantics

Category Theory

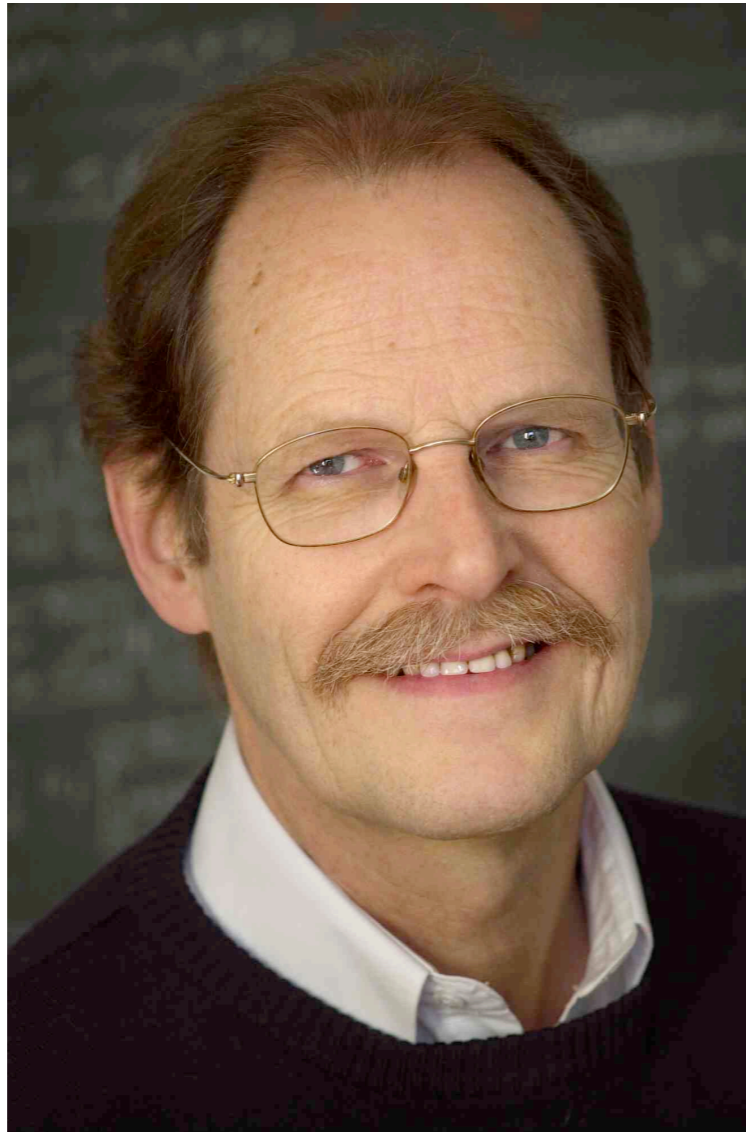
Functional Programming

Type Theory

Static Analysis

**The Mathematical Foundations of
Programming Semantics**

Semantics-Based Compiler Generation



Neil D. Jones

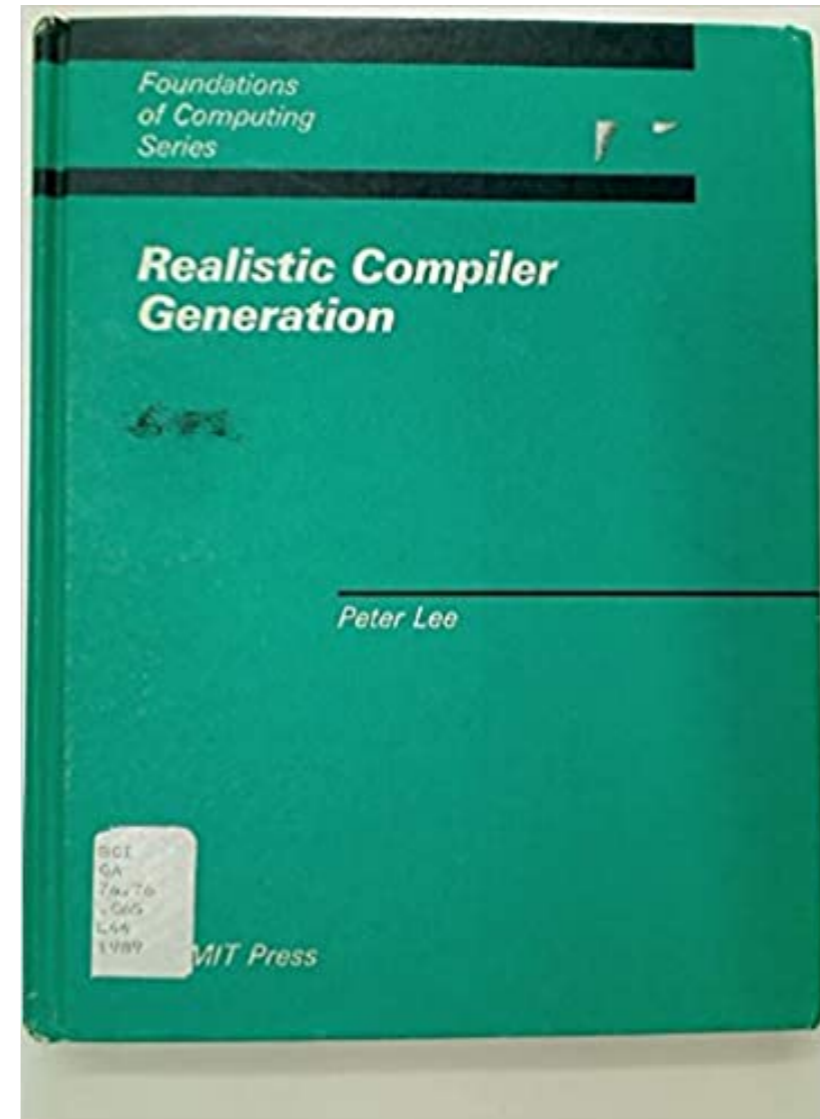


David A. Schmidt

Compiler Generation from Denotational Semantics

Semantics-Based Compiler Generation

Peter Lee



Programming Language Designer's Workbench

Semantics-Based Compiler Generation



Peter D. Mosses

Abstract Semantic Algebra



Action Semantics

Semantics-Based Compiler Generation

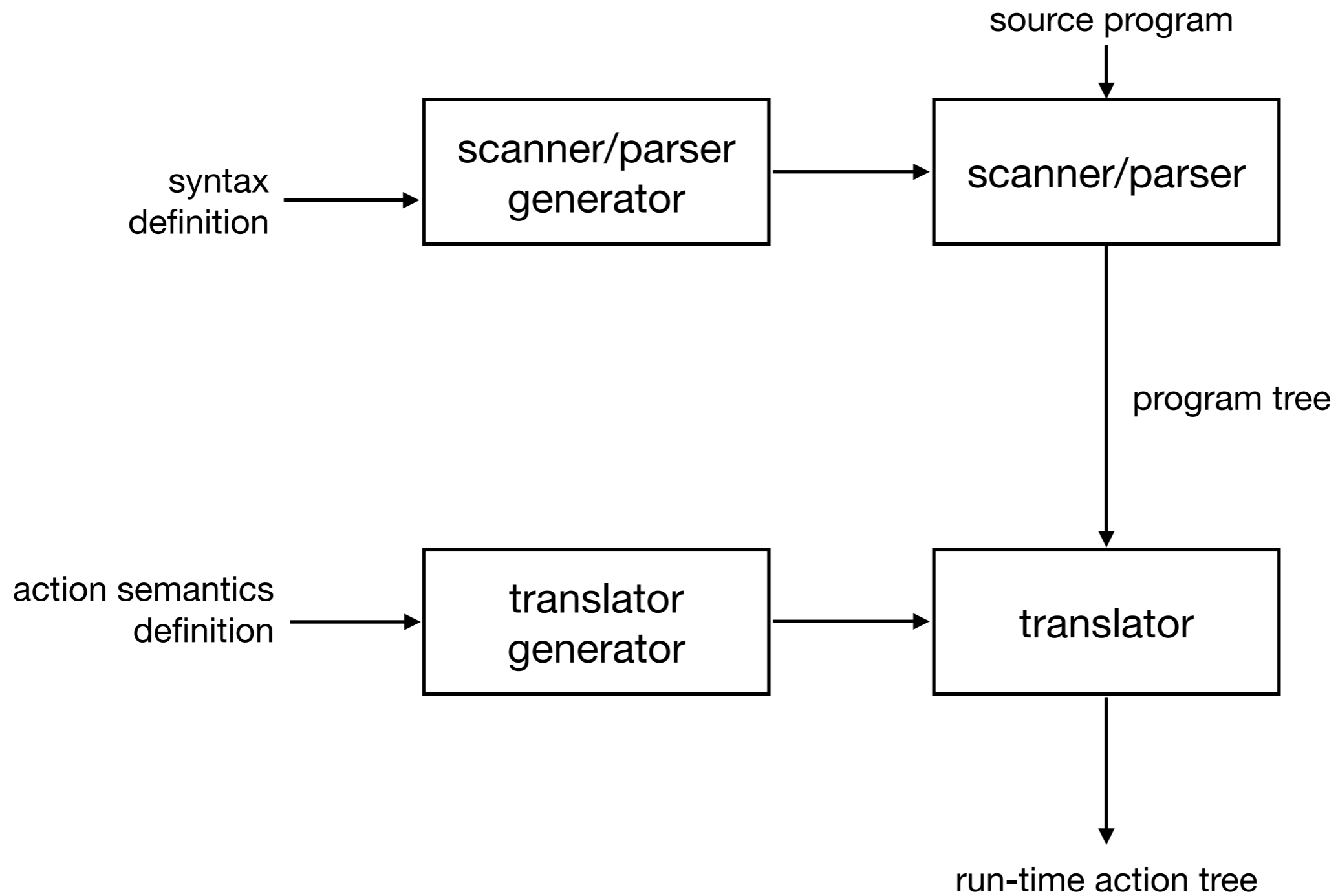


Jens Palsberg

Provably Correct Compiler Generation

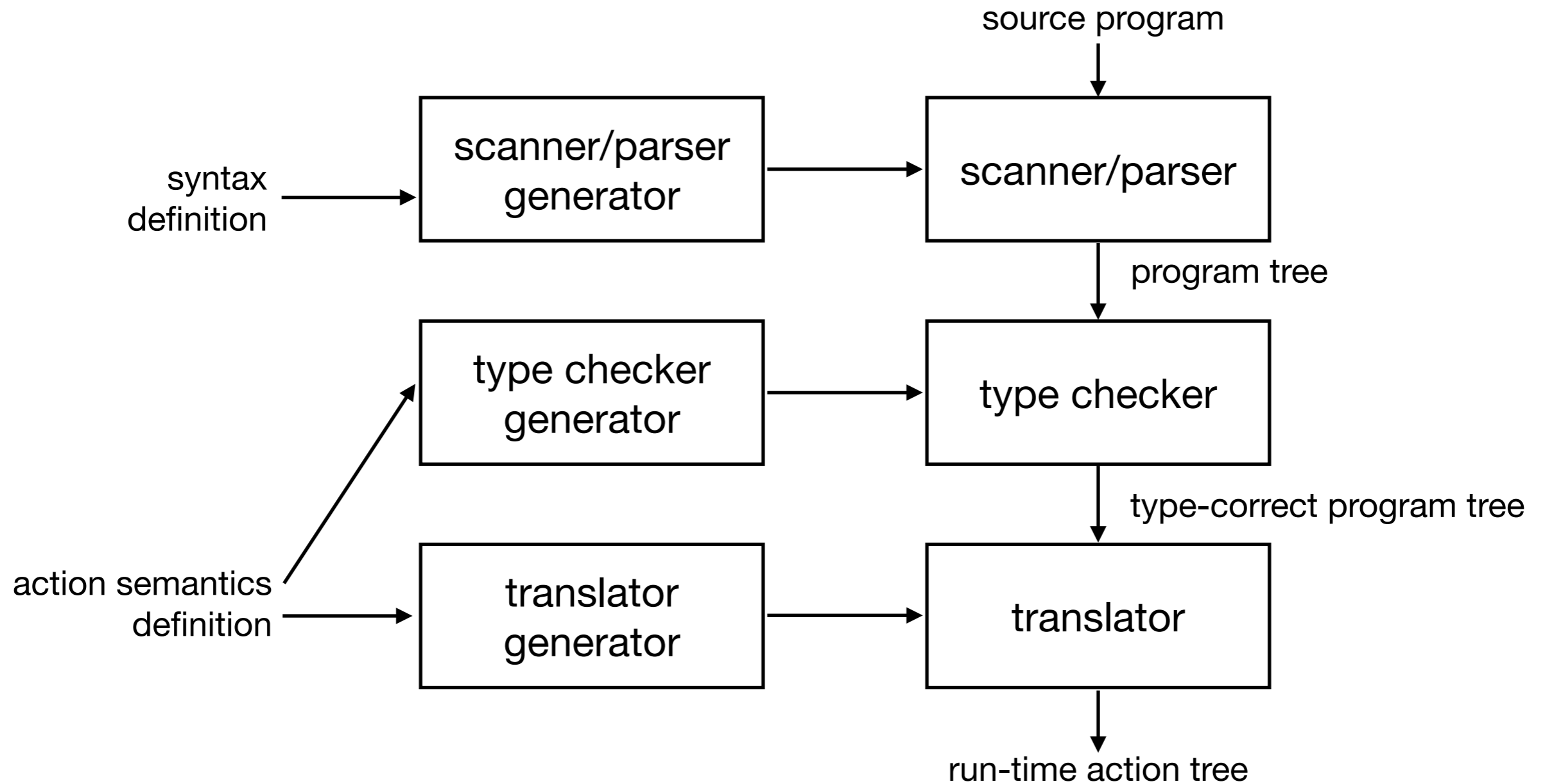
PhD Dissertation

Action Semantics-Directed Prototyping



PhD Dissertation

Action Semantics-Directed Prototyping





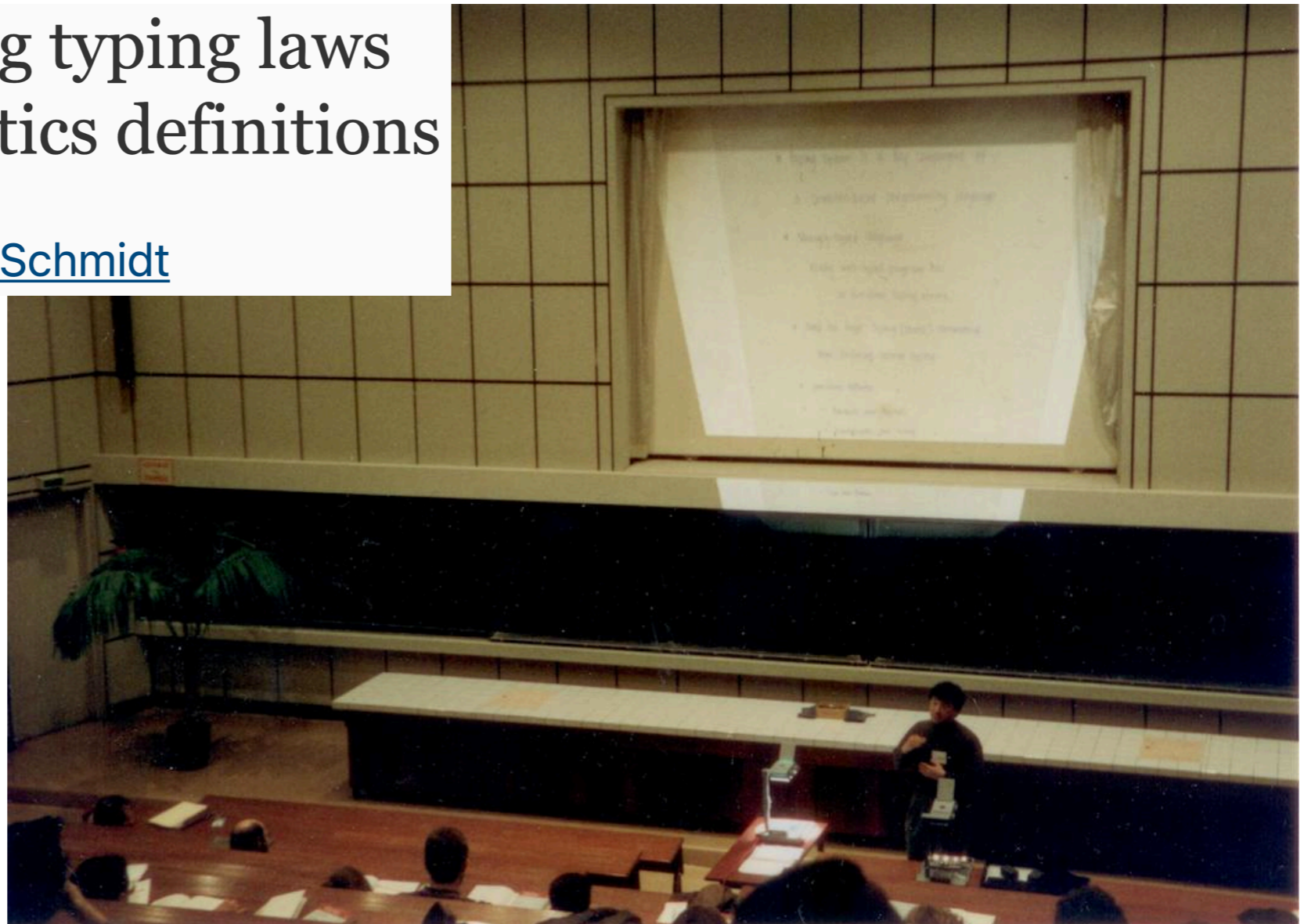
Conference proceedings | © 1992

ESOP '92

4th European Symposium on Programming, Rennes, France, February 26-28, 1992. Proceedings

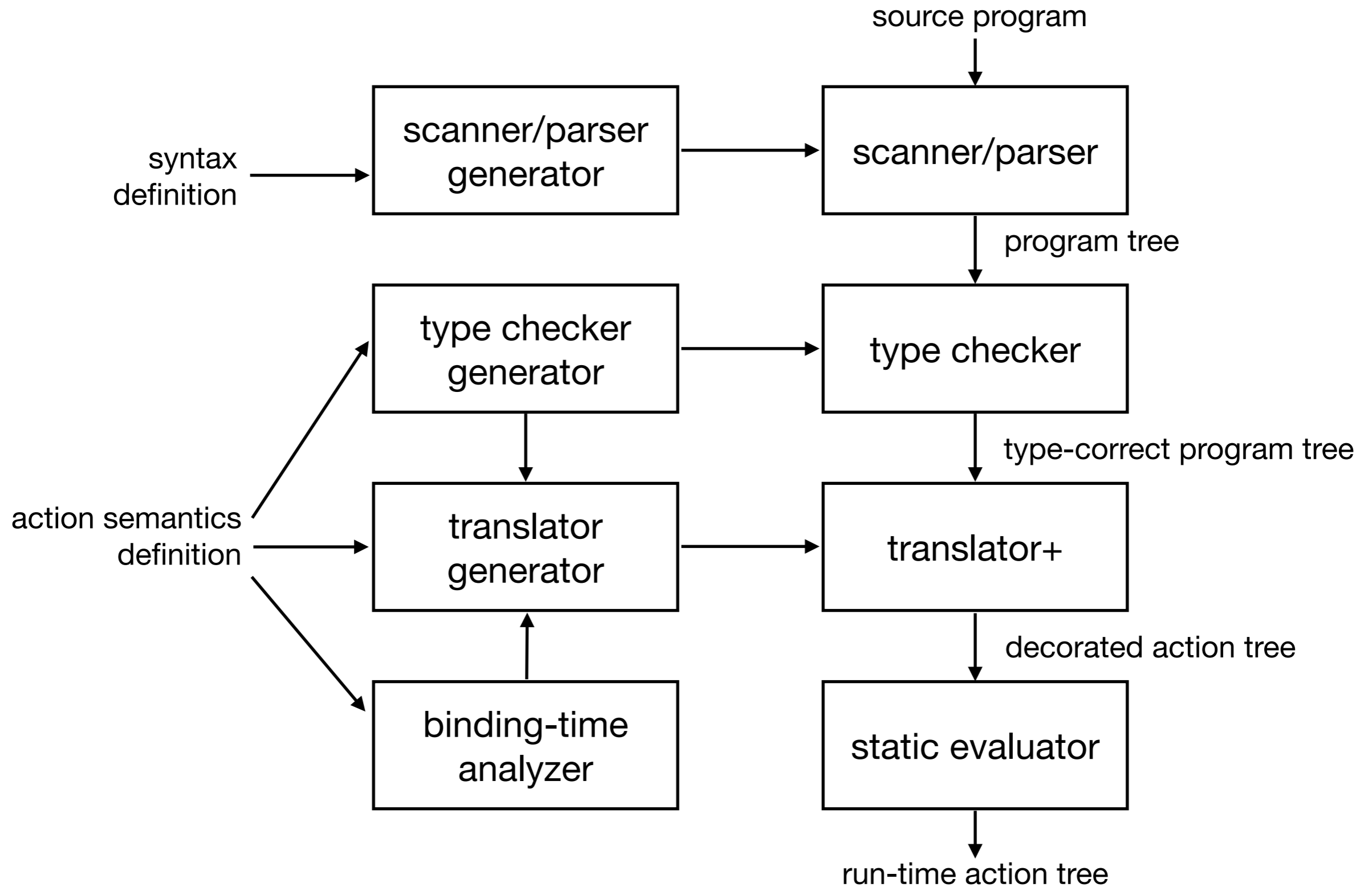
Extraction of strong typing laws from action semantics definitions

[Kyung-Goo Doh](#) & [David A. Schmidt](#)



PhD Dissertation

Action Semantics-Directed Prototyping





Action semantics-directed prototyping ☆

Kyung-Goo Doh¹, David A Schmidt²

Abstract

We present a methodology for compiler synthesis based on Mosses-Watt's action semantics. Each action in action semantics notation is assigned specific “analysis functions”, such as a typing function and a binding-time function. When a language is given an action semantics, the typing and binding-time functions for the individual actions compose into typing and binding-time analyses for the language; these are implemented as the type checker and static semantics processor, respectively, in the synthesized compiler. Other analyses can be similarly formalized and implemented. We show a sample language semantics and its synthesized compiler, and we describe the compiler synthesizer that we have developed.

첫 직장

1993년 4월



Language Processing Laboratory

Harvey Abramson

David Wei



Toshiyasu Kunii



귀국

1995년 9월

ACM Transactions on Programming Languages and Systems

ARTICLE

OPEN ACCESS

January 1985

A new analysis of LALR formalisms

 [Joseph C.H. Park](#),  [K. M. Choe](#),  [C. H. Chang](#)

ACM Transactions on Programming Languages and Systems (TOPLAS), Volume 7, Issue 1 • Jan. 1985, pp 159–175 • <https://doi.org/10.1145/2363.2527>

The traditional LALR analysis is reexamined using a new operator and an associated graph. An improved method that allows factoring out a crucial part of the computation for defining states of LR(0) canonical collection...

ACM Transactions on Programming Languages and Systems

ARTICLE

OPEN ACCESS

January 1985

A new analysis of LALR formalisms

 [Joseph C.H. Park](#),  [K. M. Choe](#),  [C. H. Chang](#)

ACM Transactions on
Systems (TOPLAS), V
175 • <https://doi.org>

The traditional LALR
operator and an asso
that allows factoring
for defining states of



ages and
n. 1985, pp 159–

ed using a new
proved method
the computation
ection...

프로그래밍언어연구회 정기총회 및 학술대회

1996년 11월

홍익대학교

PL2401

The First Asian Symposium on Programming Languages and Systems



과학재단 목적기초

프로그램 안전성을 위한 정형증명 기술

2000.9~2003.8

정주희(책임), 도경구, 배민오, 변석우, 최진영

과학재단 목적기초

프로그램 안전성을 위한 정형증명 기술

2000.9~2003.8

정주희(책임), 도경구, 배민오, 변석우, 최진영

+

이광근, 신승철

과학재단 목적기초

프로그램 안전성을 위한 정형증명 기술

2000.9~2003.8

정주희(책임), 도경구, 배민오, 변석우, 최진영

+

이광근, 신승철



LiComR

| 날짜 | 행사명 | 장소 |
|---------------|-----------------------------|---------------|
| 2023.02.22-24 | SIGPL Winter School 2023 | 서울특별시 고려대학교 |
| 2022.08.22-24 | SIGPL Summer School 2022 | 광주광역시 전남대학교 |
| 2022.02.12-14 | SIGPL Winter School 2022 | 제주 더그랜드섬오름호텔 |
| 2019.08.26-28 | SIGPL Summer School 2019 | 평창 휘닉스파크 |
| 2019.02.18-20 | SIGPL Winter School 2019 | 경성대학교 |
| 2018.08.20-22 | SIGPL Summer School 2018 | 동국대학교 |
| 2018.02.19-21 | SIGPL Winter School 2018 | 충남대학교 |
| 2017.02.8-10 | SIGPL Winter School 2017 | KAIST |
| 2016.08.17-19 | SIGPL Summer School 2016 | 광주과학기술원 |
| 2015.08.19-22 | SIGPL Summer School 2015 | 동아대학교 |
| 2013.08.24-27 | SIGPL Summer School 2013 | 송광사 |
| 2012.02.02-04 | SIGPL Winter School 2012 | 경북 영주 무섬마을 |
| 2010.02.19 | SIGPL Winter School 2010 | 성신여자대학교 |
| 2009.12.17 | Secure Coding Standard 단기강좌 | 동국대학교 |
| 2009.8.18-20 | SIGPL Summer School 2009 | 목포대학교 |
| 2009.2.9-11 | SIGPL Winter School 2009 | 한림대학교 |
| 2008.8.19-21 | SIGPL Summer School 2008 | 경성대학교 |
| 2008.1.30-2.1 | SIGPL Winter School 2008 | KAIST 전산학동 |
| 2007.2.8-10 | SIGPL Winter School 2007 | 천안 국립중앙청소년수련원 |
| 2006.6.23 | SIGPL Summer School 2006 | 용평리조트 |
| 2006.2.13-15 | SIGPL Winter School 2006 | 보광휘닉스파크 |
| 2005.7.8 | SIGPL Summer School 2005 | 보광휘닉스파크 |
| 2005.2.17-19 | SIGPL Winter School 2005 | 순천향대학교 |
| 2004.8.11-13 | SIGPL Summer School 2004 | 부산대학교 |
| 2004.2.11-13 | LiComR Winter School 2004 | 만리포해양연수원 |
| 2003.8.18-21 | LiComR Summer 2003 | 송광사 |

| 날짜 | 행사명 | 장소 |
|---------------|-----------------------------|---------------|
| 2023.02.22-24 | SIGPL Winter School 2023 | 서울특별시 고려대학교 |
| 2022.08.22-24 | SIGPL Summer School 2022 | 광주광역시 전남대학교 |
| 2022.02.12-14 | SIGPL Winter School 2022 | 제주 더그랜드섬오름호텔 |
| 2019.08.26-28 | SIGPL Summer School 2019 | 평창 휘닉스파크 |
| 2019.02.18-20 | SIGPL Winter School 2019 | 경성대학교 |
| 2018.08.20-22 | SIGPL Summer School 2018 | 동국대학교 |
| 2018.02.19-21 | SIGPL Winter School 2018 | 충남대학교 |
| 2017.02.8-10 | SIGPL Winter School 2017 | KAIST |
| 2016.08.17-19 | SIGPL Summer School 2016 | 광주과학기술원 |
| 2015.08.19-22 | SIGPL Summer School 2015 | 동아대학교 |
| 2013.08.24-27 | SIGPL Summer School 2013 | 송광사 |
| 2012.02.02-04 | SIGPL Winter School 2012 | 경북 영주 무섬마을 |
| 2010.02.19 | SIGPL Winter School 2010 | 성신여자대학교 |
| 2009.12.17 | Secure Coding Standard 단기강좌 | 동국대학교 |
| 2009.8.18-20 | SIGPL Summer School 2009 | 목포대학교 |
| 2009.2.9-11 | SIGPL Winter School 2009 | 한림대학교 |
| 2008.8.19-21 | SIGPL Summer School 2008 | 경성대학교 |
| 2008.1.30-2.1 | SIGPL Winter School 2008 | KAIST 전산학동 |
| 2007.2.8-10 | SIGPL Winter School 2007 | 천안 국립중앙청소년수련원 |
| 2006.6.23 | SIGPL Summer School 2006 | 용평리조트 |
| 2006.2.13-15 | SIGPL Winter School 2006 | 보광휘닉스파크 |
| 2005.7.8 | SIGPL Summer School 2005 | 보광휘닉스파크 |
| 2005.2.17-19 | SIGPL Winter School 2005 | 순천향대학교 |
| 2004.8.11-13 | SIGPL Summer School 2004 | 부산대학교 |
| 2004.2.11-13 | LiComR Winter School 2004 | 만리포해양연수원 |
| 2003.8.18-21 | LiComR Summer 2003 | 송광사 |



| 날짜 | 행사명 | 장소 |
|---------------|-----------------------------|---------------|
| 2023.02.22-24 | SIGPL Winter School 2023 | 서울특별시 고려대학교 |
| 2022.08.22-24 | SIGPL Summer School 2022 | 광주광역시 전남대학교 |
| 2022.02.12-14 | SIGPL Winter School 2022 | 제주 더그랜드섬오름호텔 |
| 2019.08.26-28 | SIGPL Summer School 2019 | 평창 휘닉스파크 |
| 2019.02.18-20 | SIGPL Winter School 2019 | 경성대학교 |
| 2018.08.20-22 | SIGPL Summer School 2018 | 동국대학교 |
| 2018.02.19-21 | SIGPL Winter School 2018 | 충남대학교 |
| 2017.02.8-10 | SIGPL Winter School 2017 | KAIST |
| 2016.08.17-19 | SIGPL Summer School 2016 | 광주과학기술원 |
| 2015.08.19-22 | SIGPL Summer School 2015 | 동아대학교 |
| 2013.08.24-27 | SIGPL Summer School 2013 | 송광사 |
| 2012.02.02-04 | SIGPL Winter School 2012 | 경북 영주 무섬마을 |
| 2010.02.19 | SIGPL Winter School 2010 | 성신여자대학교 |
| 2009.12.17 | Secure Coding Standard 단기강좌 | 동국대학교 |
| 2009.8.18-20 | SIGPL Summer School 2009 | 목포대학교 |
| 2009.2.9-11 | SIGPL Winter School 2009 | 한림대학교 |
| 2008.8.19-21 | SIGPL Summer School 2008 | 경성대학교 |
| 2008.1.30-2.1 | SIGPL Winter School 2008 | KAIST 전산학동 |
| 2007.2.8-10 | SIGPL Winter School 2007 | 천안 국립중앙청소년수련원 |
| 2006.6.23 | SIGPL Summer School 2006 | 용평리조트 |
| 2006.2.13-15 | SIGPL Winter School 2006 | 보광휘닉스파크 |
| 2005.7.8 | SIGPL Summer School 2005 | 보광휘닉스파크 |
| 2005.2.17-19 | SIGPL Winter School 2005 | 순천향대학교 |
| 2004.8.11-13 | SIGPL Summer School 2004 | 부산대학교 |
| 2004.2.11-13 | LiComR Winter School 2004 | 만리포해양연수원 |
| 2003.8.18-21 | LiComR Summer 2003 | 송광사 |

과학재단 목적기초

프로그램 안전성을 위한 정형증명 기술

2000.9~2003.8

정주희(책임), 도경구, 배민오, 변석우, 최진영



Software Security

과학재단 목적기초

프로그램 안전성을 위한 정형증명 기술

2000.9~2003.8

정주희(책임), 도경구, 배민오, 변석우, 최진영



Detection of information leak by data flow analysis

Authors:  [Kyung Goo Doh](#),  [Seung Cheol Shin](#) [Authors Info & Claims](#)

ACM SIGPLAN Notices, Volume 37, Issue 8 • August 2002 • pp 66–71 • <https://doi.org/10.1145/596992.597005>

과학재단 목적기초

프로그램 안전성을 위한 정형증명 기술

2000.9~2003.8

정주희(책임), 도경구, 배민오, 변석우, 최진영



명령형 프로그램의 핵심부분에 대한 정보흐름 보안성의 데이터 흐름 분석

Data Flow Analysis of Secure Information-Flow in Core Imperative Programs

정보과학회논문지 : 소프트웨어 및 응용

2004, vol.31, no.5, pp. 667-676 (10 pages)

신승철¹, 변석우 / Sugwoo Byun², 정주희³, 도경구 / Doh, Kyung-Goo⁴

¹동양대학교

²경성대학교

³경북대학교

⁴한양대학교

과학재단 목적기초

프로그램 안전성을 위한 정형증명 기술

2000.9~2003.8

정주희(책임), 도경구, 배민오, 변석우, 최진영



명령형 프로그램의 핵심부분에 대한 정보흐름 보안성의 데이터 흐름 분석

Data Flow Analysis of Secure Information-Flowin Core Imperative Programs

정보과학회논문지 : 소프트웨어 및 응용

2004, vol.31, no.5, pp. 667-676 (10 pages)

신승철¹, 변석우 /Sugwoo Byun², 정주희³, 도경구 /Doh, Kyung-Goo⁴

¹동양대학교

²경성대학교

³경북대학교

⁴한양대학교

Sindoh 가헌신도재단

가헌학술상



PL Designer's Workbench 연구 계속

정보통신부 대학기초 (1996.7~1999.6)

과학재단 핵심전문 (1997.3~1999.2)



Science of Computer Programming

Volume 47, Issue 1, April 2003, Pages 3-36



Composing programming languages by combining action-semantics modules

Kyung-Goo Doh^{a 1}  , Peter D. Mosses^{b 2} 

산업자원부 + IC카드연구조합 + 효성(주) + 아이티플러스(주) + ...

IC카드를 활용한 개방형 전자화폐시스템

(1997)~1999~2004

산업자원부 + IC카드연구조합 + 효성(주) + 아이티플러스(주) + ...

IC카드를 활용한 개방형 전자화폐시스템

(1997)~1999~2004



아이티플러스(주)

String Analysis

2003.10~2008.11

Universal String Analyzer 특허

String Analysis



Asian Symposium on Programming Languages and Systems

↳ APLAS 2006: **Programming Languages and Systems** pp 374–388 | Cite as

A Practical String Analyzer by the Widening Approach

Tae-Hyoung Choi, Oukseh Lee, Hyunha Kim & Kyung-Goo Doh

Abstract Parsing



International Static Analysis Symposium

↳ SAS 2009: **Static Analysis** pp 256–272 | Cite as

Abstract Parsing: Static Analysis of Dynamically Generated String Output Using LR-Parsing Technology

[Kyung-Goo Doh](#), [Hyunha Kim](#) & [David A. Schmidt](#)



International Static Analysis Symposium

↳ SAS 2013: **Static Analysis** pp 194–214 | Cite as

Static Validation of Dynamically Generated HTML Documents Based on Abstract Parsing and Semantic Processing

[Hyunha Kim](#), [Kyung-Goo Doh](#) & [David A. Schmidt](#)

과학기술부 특정기초

유비쿼터스 컴퓨팅을 위한 프로그래밍 환경 연구

2006.3~2009.2

창병모(책임), 도경구, 안준선



International Conference on Embedded and Ubiquitous Computing
↳ EUC 2006: **Emerging Directions in Embedded and Ubiquitous Computing**
pp 650–659 | [Cite as](#)

A Policy Description Language for Context-Based Access Control and Adaptation in Ubiquitous Environment

[Joonseon Ahn](#), [Byeong-Mo Chang](#) & [Kyung-Goo Doh](#)

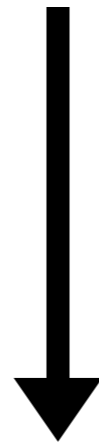
A programming environment for ubiquitous computing environment

Authors:  [Minkyung Oh](#),  [Jiyeon Lee](#),  [Byeong-Mo Chang](#),  [Joonseon Ahn](#),  [Kyung-Goo Doh](#)

아이티플러스(주)

Security Analysis

2006.9~2007.8



행정안전부 정보시스템 보안성강화체계 구축 사업 (KISA)

정보보호학회 소프트웨어보안연구회, 지티원, 파수닷컴

소프트웨어 취약성 점검도구 및 보안성 강화 지원시스템 개발

2009.9~2010.12

한근희, 최진영, 오세만, 도경구, 이광근, 창병모, 안준선, 한경숙, 신승철, ...

교육과학기술부 ERC

소프트웨어 무결점 연구센터
ROSAEC

센터장 이광근 교수

2008.9~2014.2



교육

SW특성화대학 · 2012~2016

소프트웨어융합대학 신설 · 2017~

SW중심대학 · 2018~

교육

SW특성화대학 · 2012~2016

소프트웨어융합대학 신설 · 2017~

SW중심대학 · 2018~

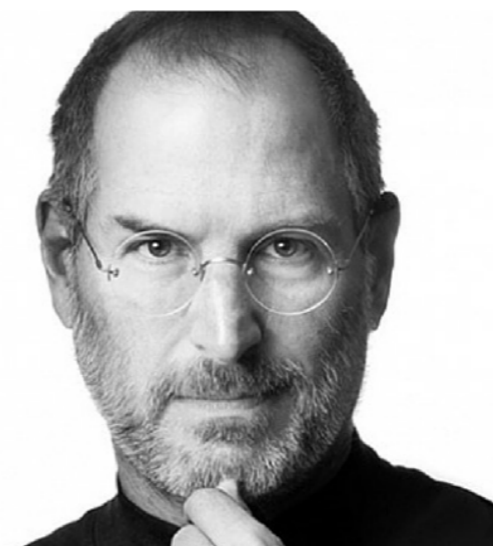
Forbes /

NOV 30, 2011 @ 01:58 PM 50,625 VIEWS

Now Every Company Is A Software Company

Everybody in this
country should
learn to program a
computer, because
it teaches you how
to think

- Steve Jobs -



음악 프로그래밍

Chuck

**Strongly-timed,
Concurrent,
On-the-fly
Music Programming Language**



Perry R. Cook



Ge Wang



miniAudicle

version 1.4.1.0 (gidora)

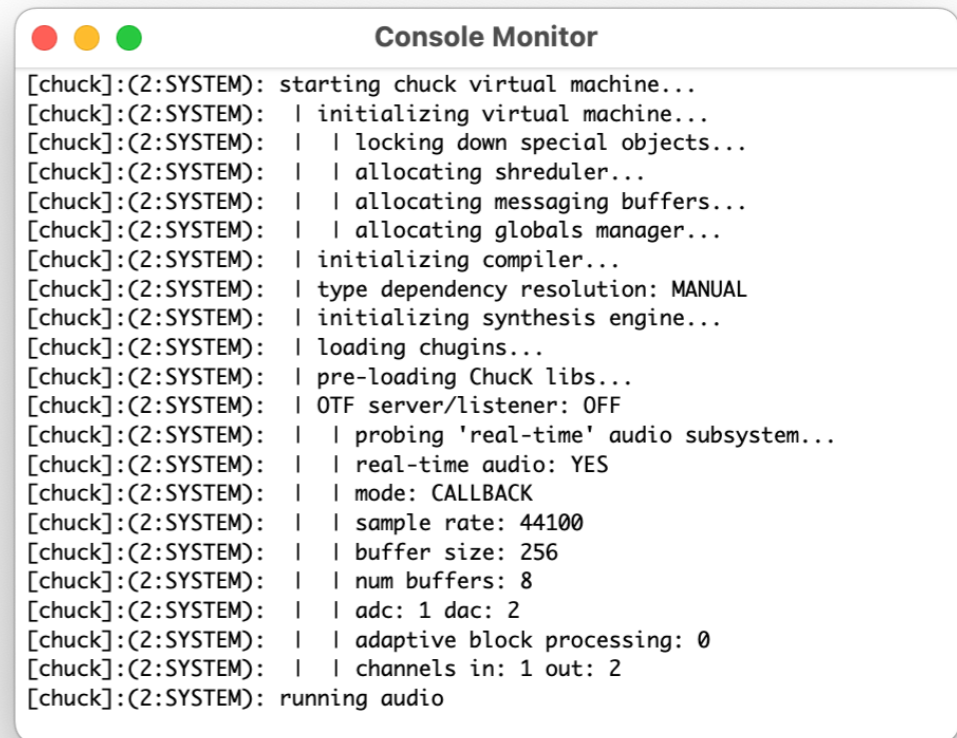
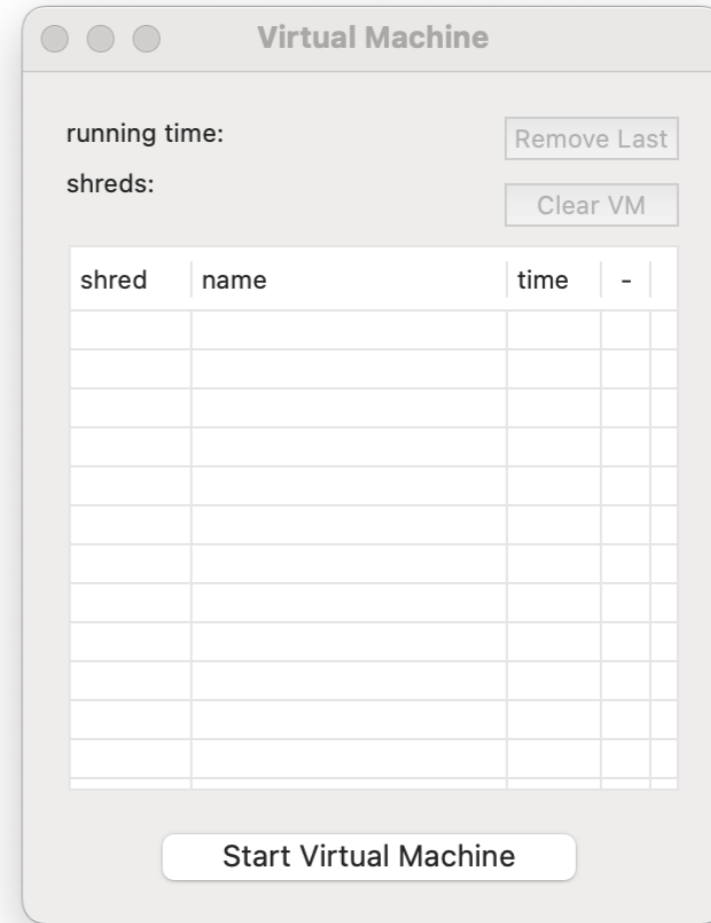
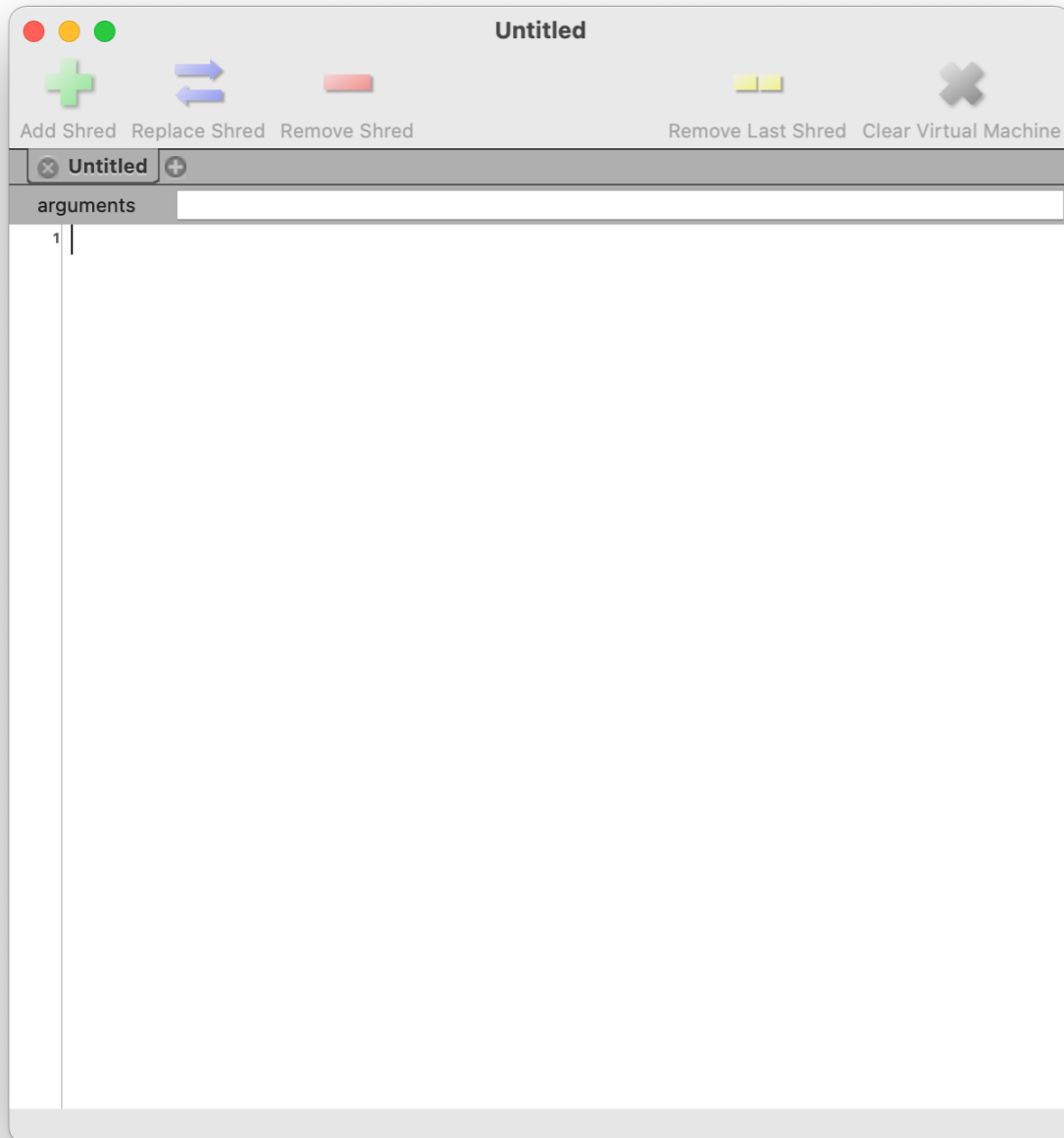
git: 9f6ee78

Copyright (c) Spencer Salazar

ChuckK: version 1.4.1.0 (numchucks) 64-bit

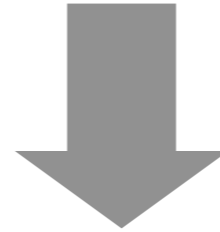
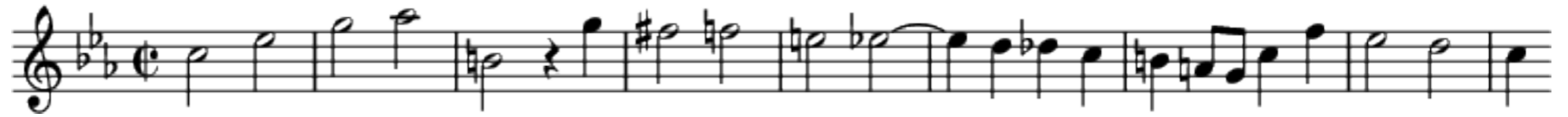
Copyright (c) Ge Wang and Perry Cook

<http://chuck.cs.princeton.edu/>

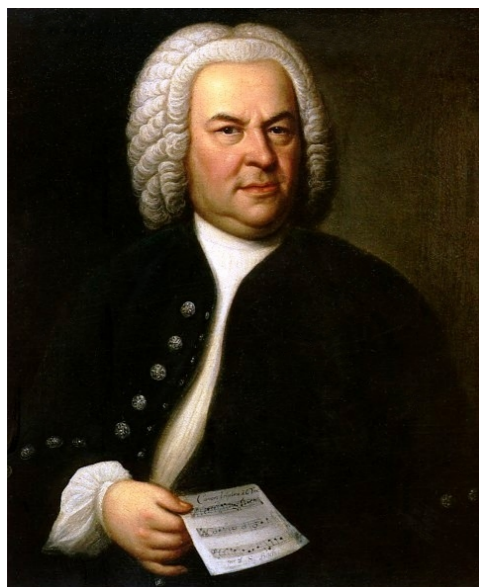


Demo

**King Frederick
The Royal Theme**



**Johann Sebastian Bach
Musical Offering, BWV 1079
Canones diversi super thema regium: Canon a 2
“Crab Canon”**



Algorithmic Music Composition

컴퓨터를 악기로 ~

교육

SW특성화대학 · 2012~2016

소프트웨어융합대학 신설 · 2017~

SW중심대학 · 2018~

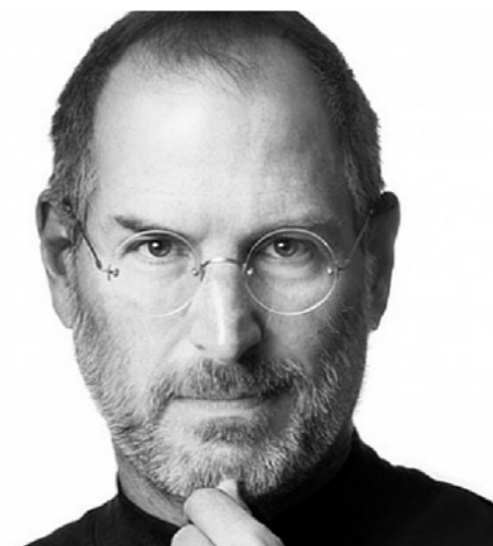
Forbes /

NOV 30, 2011 @ 01:58 PM 50,625 VIEWS

Now Every Company Is A Software Company

Everybody in this
country should
learn to program a
computer, because
it teaches you how
to think

- Steve Jobs -





교육

SW특성화대학 · 2012~2016

소프트웨어융합대학 신설 · 2017~

SW중심대학 · 2018~

Forbes /

NOV 30, 2011 @ 01:58 PM 50,625 VIEWS

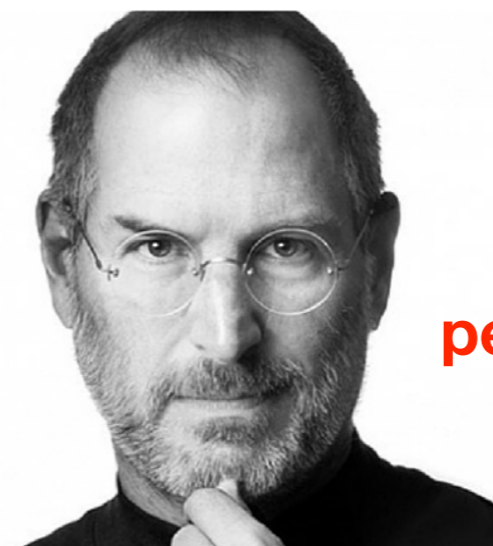
Now Every Company Is A Software Company

use

Everybody in this country should learn to ~~program~~ a computer, ~~because it teaches you how to think~~

period!

- Steve Jobs -





교육

SW특성화대학 · 2012~2016

소프트웨어융합대학 신설 · 2017~

SW중심대학 · 2018~

소프트웨어는

안전하고

견고하고

신뢰할 수 있는

명품 이어야 한다.

제대로 훈련받은 재능있는

명인 만이 명품을 만들 수 있다.



Domain-Specific Language Designer

POPL 2012

Defining Code-injection Attacks

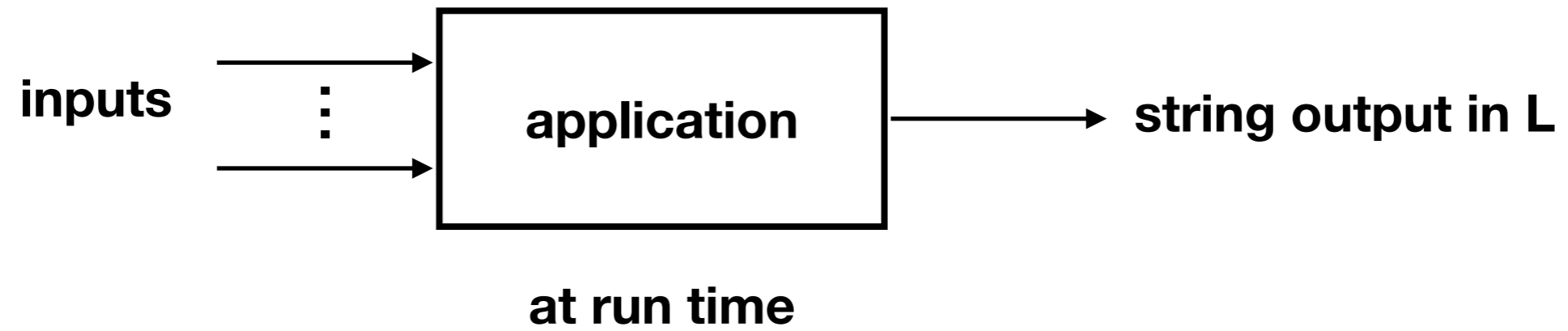
Donald Ray Jay Ligatti

ISC 2014

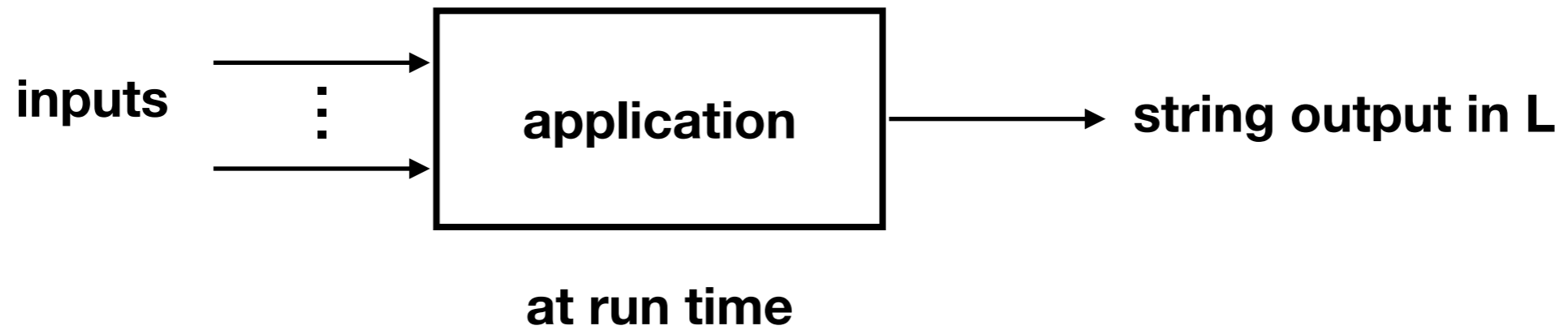
Defining Injection Attacks

Donald Ray and Jay Ligatti

Secure Injection Principles



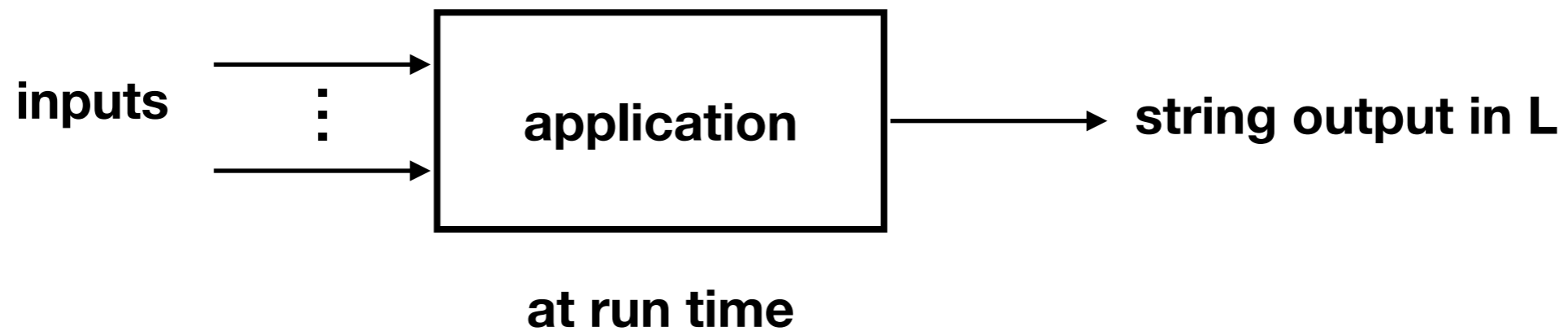
Secure Injection Principles



Principle 1



Secure Injection Principles



Principle 1



Principle 2

any input injected elsewhere should be

- a syntactically legal and fully evaluated term in L and
- only injected to be an expression

Theorem

The output-assembling program is written fully in compliance with the secure injection principle.



Injection attacks defined in Ray and Ligatti's papers are all incapacitated

Theorem

The output-assembling program is written fully in compliance with the secure injection principle.



Injection attacks defined in Ray and Ligatti's papers are all incapacitated

proved!

Automatic Fortification of Web Applications Against Injection Attacks

Automatic Fortification of Web Applications Against Injection Attacks

**separate functional logic
from defense logic
when writing applications**

Automatic Fortification of Web Applications Against Injection Attacks

**separate functional logic
from defense logic
when writing applications**

- **The developer only implements functional logic.**
- **Our fortification tool automatically patches defense logic.**

감사합니다!

**SIGPL과 함께 한
그동안의 여정이
행복했습니다!**

SOFTOPIA