CS6843: Program Analysis

Instructor: Rupesh Nasre (rupesh@cse)
TAs: Raghesh, Rakesh, Saurabh

Web: ~rupesh/teaching/pa/jan15/
Moodle: moodle/course/view.php?id=365

Jan 2015

What *(the hell)* is Program Analysis?

For an end-goal identify “interesting aspects” of a program's representation.

What *(the hell)* is Program Analysis?

For an end-goal

Checking security

identify “interesting aspects”

Array index range

of a program's representation.

Source, AST, binary,
executed instruction

Examples

End goal	Interesting aspect
Dead code elimination	Reachability
Constant propagation	use-def
Security	Array index range, dangling pointers
Parallelization	Data dependence, SIMD opportunities
Debugging	Slice
Cache performance	Memory access pattern
Memory reduction	Live ranges
...	...

Program Analysis is often a pre-cursor to Optimization.

In This Course

7. Security Analysis (SEC)

6. Program Slicing (SLI)

5. Parallelization (PAR)

4. Polyhedral Model (POL)

3. Dynamic Analysis (DYN)

2. Shape Analysis (SHA)

1. Pointer Analysis (PTR)

0. Data Flow Analysis (DFA)

Logistics

- Moodle for submissions, announcements, discussions
  - Your responsibility to subscribe to it.
- Evaluation:
  - assignments (25%)
  - course project (25%)
  - midsem (25%)
  - endsem (25%)
- C slot (Mon 10, Tue 9, Wed 8, Fri 12).
  - Friday slot would be mainly used for doubts + examples.
- Room CS 26.

Assignments

- Three programming assignments (25%).
- We will increase the complexity (and marks) gradually.
- Assignments would be in LLVM.
- You can submit late (within two days) but you will lose half marks. Beyond two days, you need not submit.
- You should work individually.

Course Project

- 25% marks.
- Need not be in LLVM, need not be in C.
- Sample topics are listed on the webpage, but you can choose your own after discussing with the instructor.
- It will be evaluated in two phases (CP1 and CP2) each having a presentation and a demo.
- You can work in a group of one or two.

Course Schedule

Month	Lectures	Evaluations
JAN	DFA, SLI	A1
FEB	PAR, DYN	A2, A3
MAR	PTA, SHA	MIDSEM, CP1
APR	POL, SEC	CP2
MAY	--	ENDSEM

MidSem and EndSem will have
mutually exclusive topics.

Example Three

```
void main() {
  int a, b, c, d, *p;

  p = &a;
  c = a + b;
  d = *p + b;
}
```

Can this computation be avoided?
(*common subexpression elimination*)

```
void main() {
  int a, b, c, d, *p;

  p = &a;
  int t = a + b;
  c = t;
  d = t;
}
```

This requires a program analysis called *pointer analysis*.

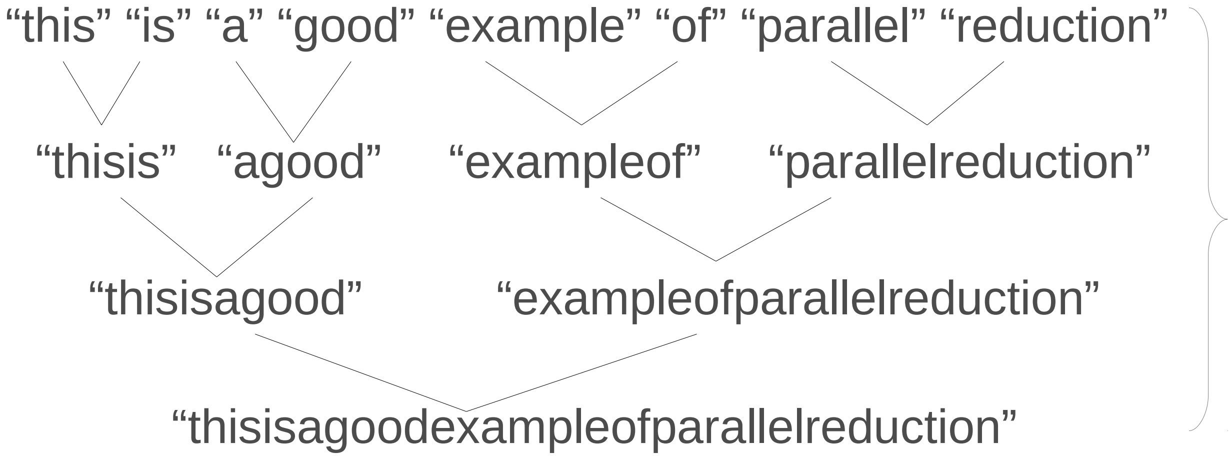
This requires another analysis called *type analysis*.

Example Two

```
*sresult = 0;

for (ii = 0; ii < nn; ++ii) {
  strcat(sresult, str[ii]);
}
```

Can you parallelize this code?



“this” “is” “a” “good” “example” “of” “parallel” “reduction”

“thisis” “agood” “exampleof” “parallelreduction”

“thisisagood” “exampleofparallelreduction”

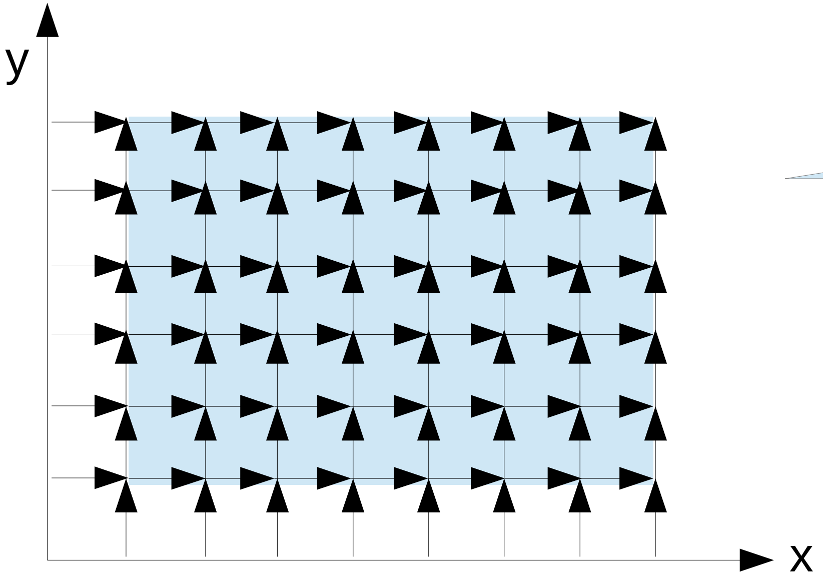
“thisisagoodexampleofparallelreduction”

Requires *semantic analysis* to figure out that *strcat* performs an **associative** operation.

Example One

```
for (x = 1; x < M; ++x)
  for (y = 1; y < N; ++y)
    a[x, y] = a[x – 1, y] + a[x, y – 1];
```

Can you parallelize iterations?



Requires
loop dependence analysis

More in polyhedral model