Introduction to COMS22201: *Language Engineering*

Steve Gregory & Oliver Ray
Department of Computer Science
University of Bristol

Monday 28th September, 2015

Lecture Contents

- Unit Management
  - unit staff and how to contact us
  - attendance and assessment
- Unit Overview
  - compiler techniques
  - semantic methods

Unit Staff

- Unit Lecturers
  - TB1: Steve Gregory steve@cs.bris.ac.uk 3.21
  - TB2: Oliver Ray csxor@bristol.ac.uk 3.29

- Lab Helpers
  - TB1: Craig Blackmore + others
  - TB2: Craig Blackmore + ?

The best ways to contact us

- We strongly encourage you to ask questions in lectures (unless your query is too personal) or in labs (unless your query is too urgent)
- Otherwise, we strongly encourage you to post (and reply!) to the unit forum (unless your query is too personal). We will monitor and intervene as appropriate. Make sure you enable notifications!
- Personal issues should normally be discussed first with your personal tutor (who will liaise with us if necessary)

Some less good ways to contact us

- Email can be a good way for us to contact you, but is often not such a good way for you to contact us (unless we ask you to do so)
- Coming to our office can sometimes be helpful but we will often have to ask you to make an appointment
- In both cases make sure to say who you are, what is the problem, which steps you have already taken, and how exactly you would like us to help
- Note that you can more easily be helped if you try to help yourself first!

Unit Attendance

- Lectures (Steve/Oliver)
  - primary method of disseminating material
  - Mon 2-3 (Chemistry LT3)
  - Fri 11-12 (Physics G42)

- Labs (Steve/Oliver + assistants)
  - primary method of providing feedback
  - Mon 4-5 (2.11, A-K odd weeks, L-Z even weeks)
  - Mon 5-6 (2.11, L-Z odd weeks, A-K even weeks)
  - you may attend either lab *if* there is space

Formative and Summative Assessment

- 20 Credit Points, 6 SAFE Components:
  - CWK1p1 (compilers) week-6 0% *
  - CWK1p2 (compilers) week-8 0% *
  - CWK1 (compilers) week-12 25%
  - CWK2p1 (semantics) week-16 0% *
  - CWK2p2 (semantics) week-18 0% *
  - CWK2 (semantics) week-23 25%
  - Exam (both) summer 50%

* Submitting the formative exercises will give you useful feedback for the assessed courseworks

Formative and Summative Assessment

- Weeks 1-12 (compilers):

Week	Submit (Fri)	Lab (Mon)
4		ANTLR
5		ANTLR
6	CWK1 Part1	Feedback on CWK1 Part1
7		Feedback on CWK1 Part1
8	CWK1 Part2	Feedback on CWK1 Part2
9		Feedback on CWK1 Part2
10		Help with CWK1
11		Help with CWK1
12		Help with CWK1

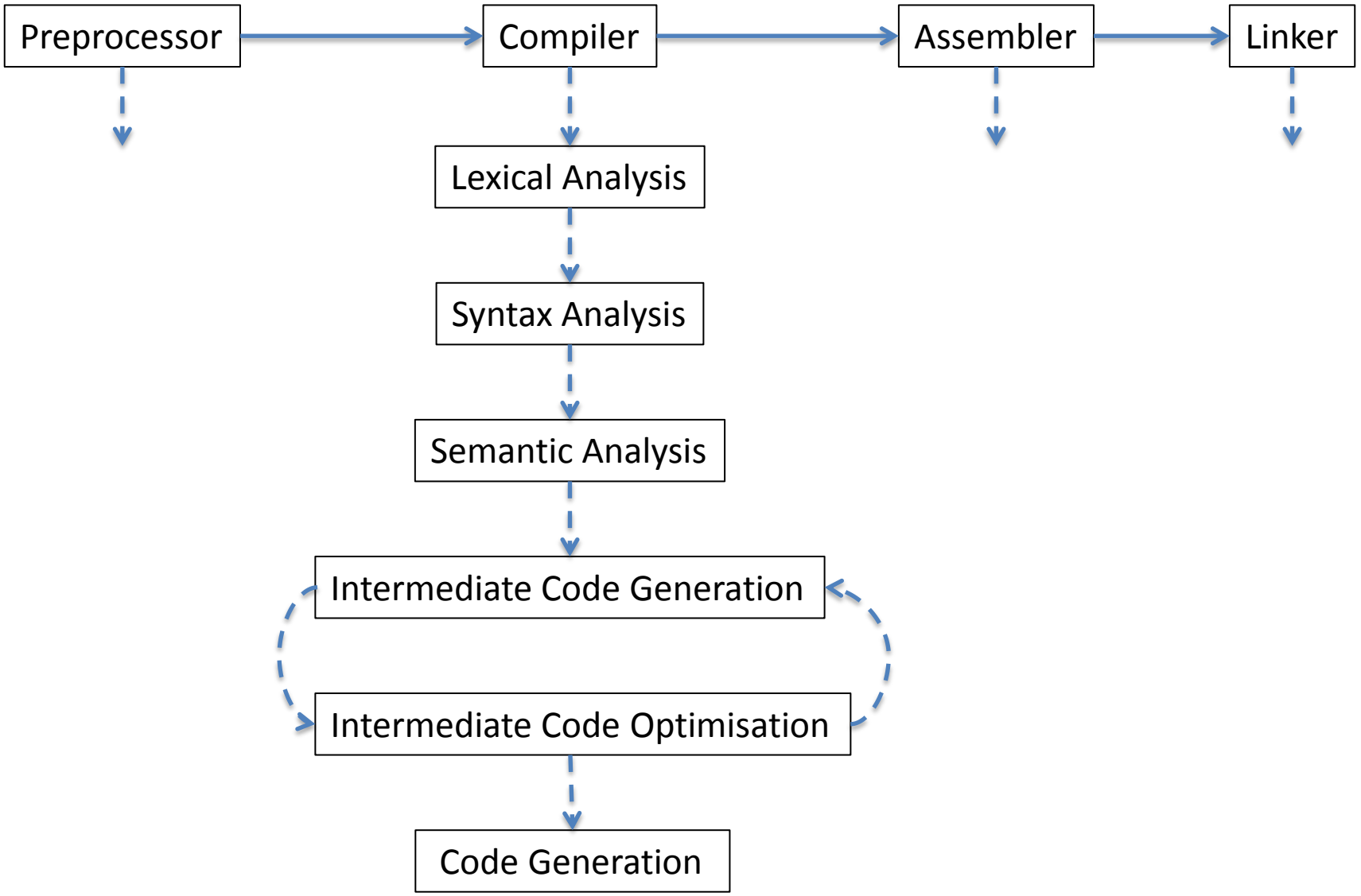
Topics Studied on the Unit

- Weeks 1-12 Compiler Techniques
  - Christmas submit CWK1
- Weeks 13-22 Semantic Methods
  - Easter submit CWK2
- Weeks 23-24 Feedback and Revision
  - Summer Exam

Learning outcomes

- Understand denotational, axiomatic, operational styles of semantics
- Formally prove the correctness of simple programs
- Parse source programs and generate low-level code from them
- Carry out analyses and optimization of programs
- Use tools to generate parsers and verify programs
- Write a complete but simple compiler

Brief Intro: Compiler Methods



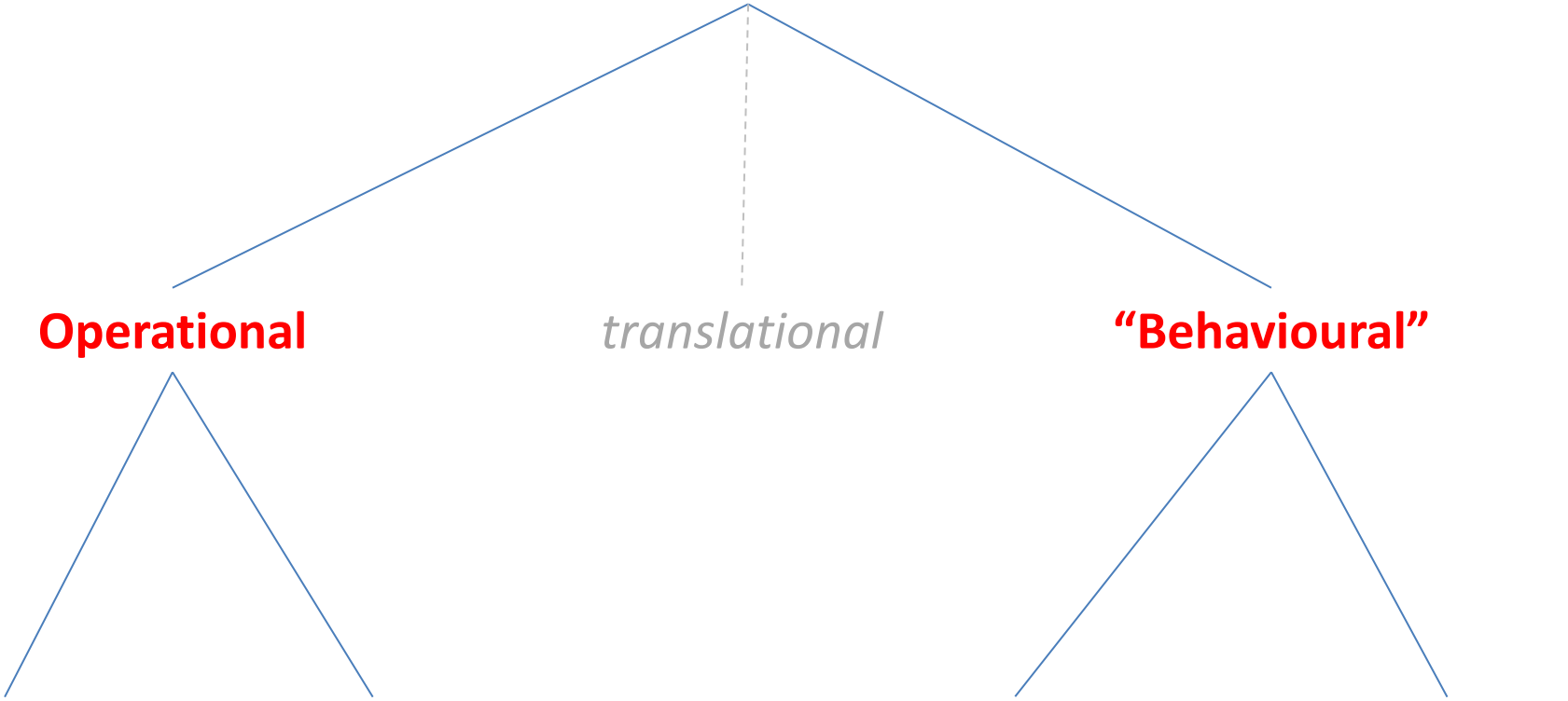
Help wanted

- I want to make coursework (and lectures) tie in with previous 1st-year units.

- I need a few volunteers to attend short meeting today at 4pm (in 3.21):
  - CS
  - CSE
  - Maths & CS

Brief Intro: Semantic Techniques

Meaning of program statements represented by mathematical functions…



Brief Intro: Links

- TB1 will help you understand and process the programming languages used in TB2.
- TB2 will help justify some of the optimizations introduced in TB1.
- TB1 will use theory and computer architecture from the 1st year.
- TB2 will exploit and develop your Haskell programming from the 1st year along with formal logic and discrete maths.

Exercise

```
// which of these C programs is fastest?
```

```
int k=0, a, b, c, d, i;
int max = 1000000000;

for (i=0; i<max; i++) {
  a = k;
  b = a;
  c = b;
  d = c;
}
printf("%d\n", d);
```

```
int k=0, d;


d = k;
printf("%d\n", d);
```

Exercise

```
// what does this GNU-C program return ?
int x = 0 ;
void p() { x *= 2 ; }
void q() { p() ; }
int main(void) {
    int x = 5 ;
    void p() { x += 1 ; }
    q() ;
    return x ;
}
```