

Automatic Compiler-Directed Software Parallelization

Adam R. Smith • asmith@ittc.ku.edu • Prasad A. Kulkarni • kulkarni@ittc.ku.edu

The Problem

- In the multi core era programs need to be parallelized to get speedup from new generations of CPUs
 - Functions must be reentrant to be parallelized
 - A reentrant function may not rely on static/global data
- Many current C programs are highly non-reentrant
 - SPEC CPU2006 examples: non-reentrant/total functions
 - bzip2 65/102, sjeng 130/144, h264ref 505/590
 - Several more examples...
- Trying to make code reentrant by hand is extremely tedious and error prone
 - Automatic compiler tools will help to more easily write parallel code

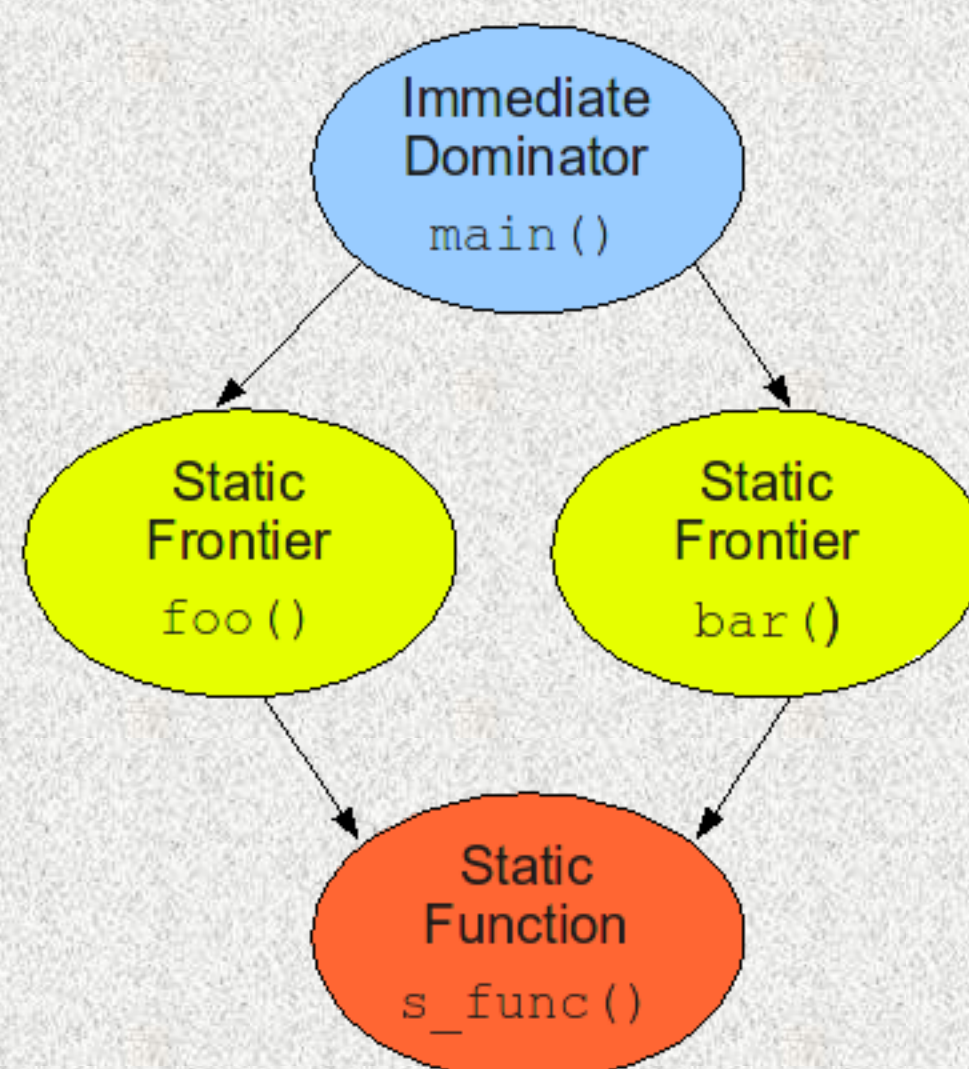
Achieving Code Reentrancy

- Algorithm
 - Input non-reentrant C code
 - Construct call graph
 - Detect statics/globals
 - Find immediate **dominator** for static/global functions
 - Convert statics/globals to locals in **dominator**
 - Change function prototypes
 - Output reentrant object files
- Non-reentrant code


```
int s_func() {
    static int s;
    return s++;
}
int foo() {
    return s_func();
}
int bar() {
    return s_func();
}
int main() {
    int i, j;
    for (i=0; i<100; i++)
        j = foo()+bar();
}
```

Achieving Code Reentrancy Cont.

• Call graph



• Reentrant code

```
int s_func(int *s) {
    return *s++;
}
int foo(int *s) {
    return s_func(s);
}
int bar(int *s) {
    return s_func(s);
}
int main() {
    int i, j;
    int s=0;
    for (i=0; i<100; i++)
        j = foo(&s)+bar(&s);
}
```

Status and Future Work

- Status
 - Currently implemented for statics
- Future Work
 - Finish implementation for globals
 - Test on real world examples
- Out tool will...
 - Make it easier to convert single threaded programs to multi threaded programs
 - Allow programmers to still use statics and globals when writing multi threaded code to simplify implementation