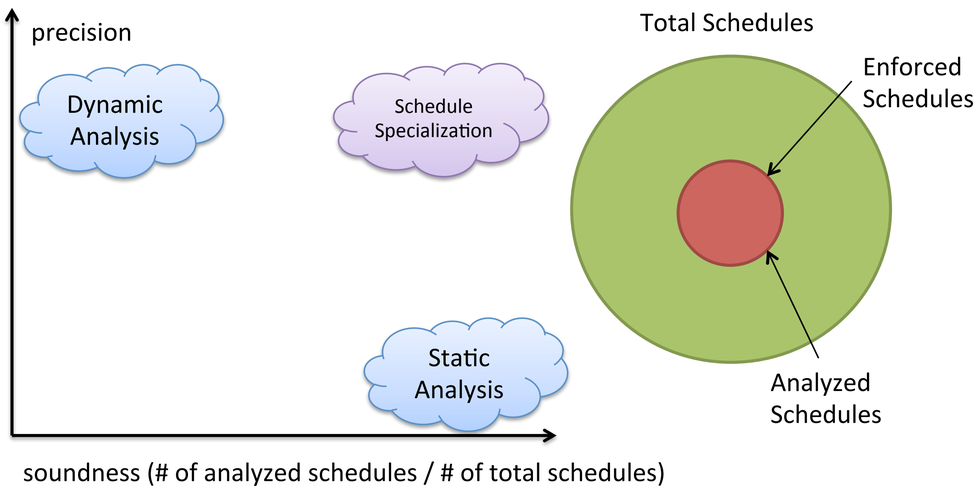
Sound and Precise Analysis of Parallel Programs through Schedule Specialization

Jingyue Wu, Yang Tang, Gang Hu, Heming Cui, Junfeng Yang
Columbia University

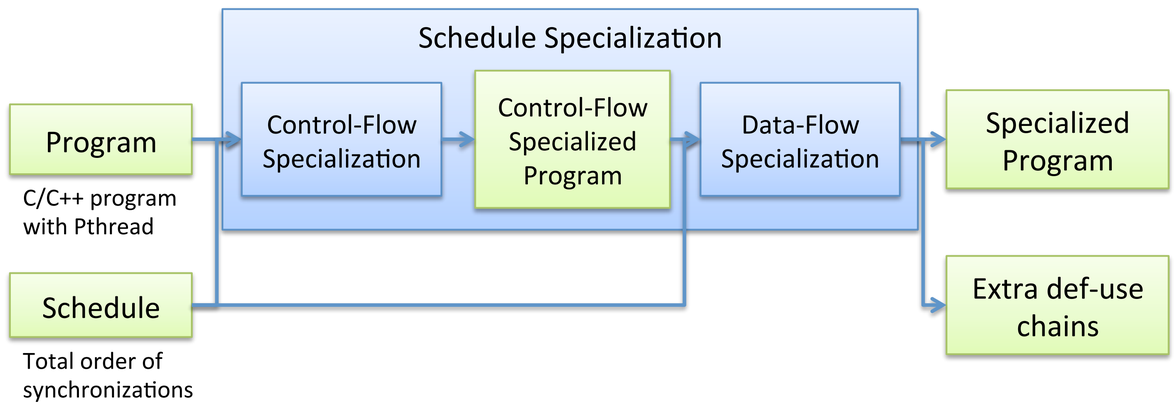
Schedule Specialization

- Precision: Analyze the program over a small set of schedules.
- Soundness: Enforce these schedules at runtime.



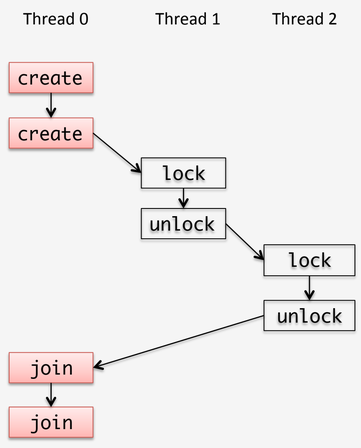
Framework

- Extract control flow and data flow enforced by a set of schedules

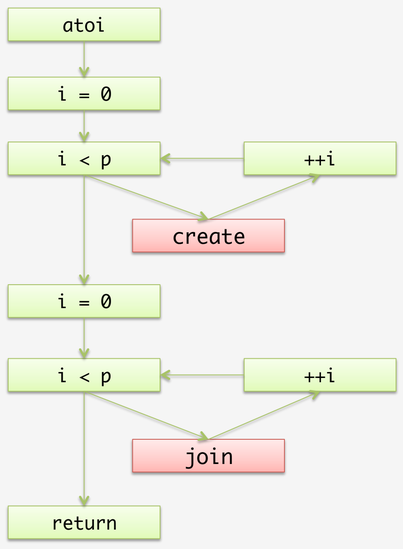



```
int results[p_max];
int global_id = 0;

int main(int argc, char *argv[]) {
  int i;
  int p = atoi(argv[1]);
  for (i = 0; i < p; ++i)
    pthread_create(&child[i], 0, worker, 0);
  for (i = 0; i < p; ++i)
    pthread_join(child[i], 0);
  return 0;
}

void *worker(void *arg) {
  pthread_mutex_lock(&global_id_lock);
  int my_id = global_id++;
  pthread_mutex_unlock(&global_id_lock);
  results[my_id] = compute(my_id);
  return 0;
}
```

Original Program

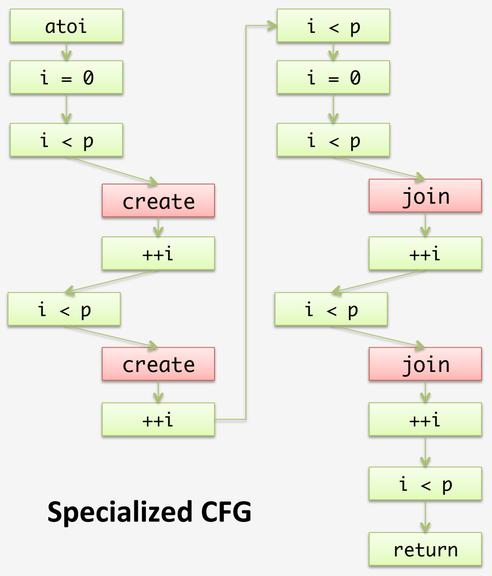


Schedule



Original CFG

Control-Flow Specialization



Specialized CFG

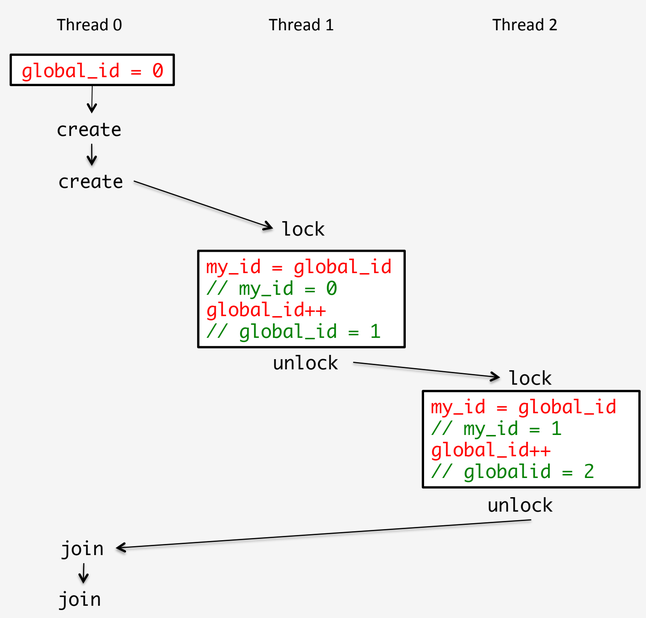
```
int main(int argc, char *argv[]) {
  int i;
  int p = atoi(argv[1]);
  i = 0; // i < p == true
  pthread_create(&child[i], 0, worker.clone1, 0);
  ++i; // i < p == true
  pthread_create(&child[i], 0, worker.clone2, 0);
  ++i; // i < p == false
  i = 0; // i < p == true
  pthread_join(child[i], 0);
  ++i; // i < p == true
  pthread_join(child[i], 0);
  ++i; // i < p == false
  return 0;
}
void *worker.clone1(void *arg) {
  pthread_mutex_lock(&global_id_lock);
  int my_id = global_id++;
  pthread_mutex_unlock(&global_id_lock);
  results[my_id] = compute(my_id);
  return 0;
}
void *worker.clone2(void *arg) {
  pthread_mutex_lock(&global_id_lock);
  int my_id = global_id++;
  pthread_mutex_unlock(&global_id_lock);
  results[my_id] = compute(my_id);
  return 0;
}
```

Control-Flow Specialized Program

Data-Flow Specialization

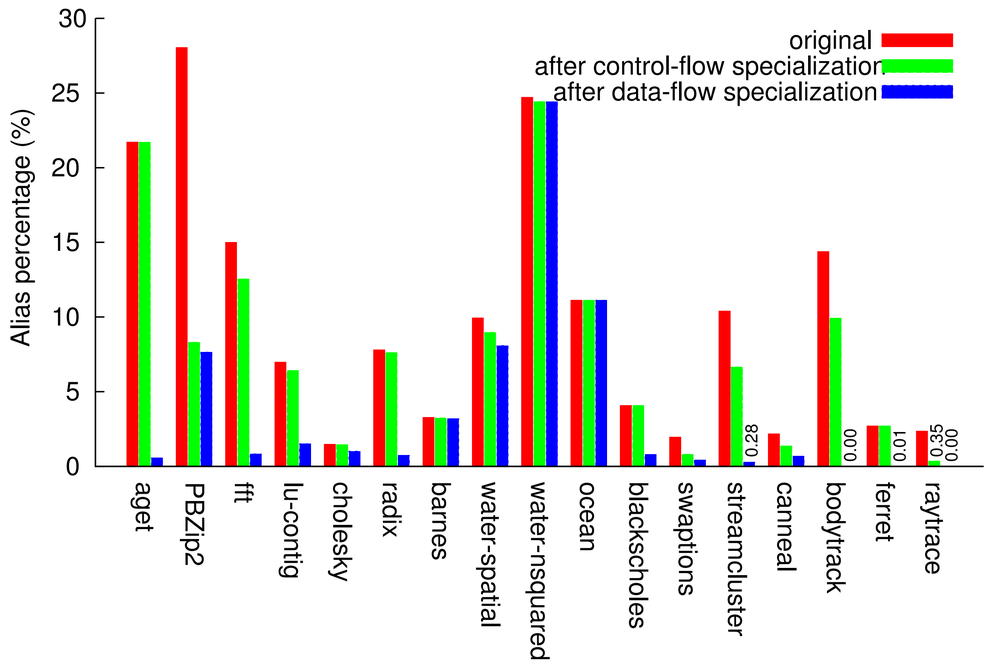
```
int main(int argc, char *argv[]) {
  int p = atoi(argv[1]);
  pthread_create(&child[0], 0, worker.clone1, 0);
  pthread_create(&child[1], 0, worker.clone2, 0);
  pthread_join(child[0], 0);
  pthread_join(child[1], 0);
  return 0;
}

void *worker.clone1(void *arg) {
  pthread_mutex_lock(&global_id_lock);
  global_id = 1;
  pthread_mutex_unlock(&global_id_lock);
  results[0] = compute(0);
  return 0;
}

void *worker.clone2(void *arg) {
  pthread_mutex_lock(&global_id_lock);
  global_id = 2;
  pthread_mutex_unlock(&global_id_lock);
  results[1] = compute(1);
  return 0;
}
```

Data-Flow Specialized Program



Data-Flow Facts

Precision of Schedule-Aware Alias Analysis



Static Race Detector

Harmful races:

- 4 in aget
- 2 in radix
- 1 in fft

# of False Positives

Program	Original	Specialized
aget	72	0
PBZip2	125	0
fft	96	0
blackscholes	3	0
swaptions	165	0
streamcluster	4	0
canneal	21	0
bodytrack	4	0
ferret	6	0
raytrace	215	0
cholesky	31	7
radix	53	14
water-spatial	2447	1799
lu-contig	18	18
barnes	370	369
water-nsquared	354	333
ocean	331	292