PFunc: A Tool For Teaching And Implementing Task Parallelism

Prabhanjan Kambadur[1], Anshul Gupta[1], Andrew Lumsdaine[2]
[1] IBM TJ Watson Research Center. [2] Indiana University, Bloomington

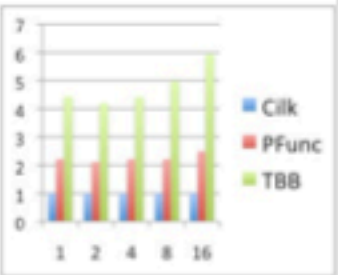
Task Parallelism

```
struct fibonacci {
 fibonacci (const int& n):(n),answer(0){}
 void operator () () {
  if (0 == n || 1 == n) answer = n;
  else {
   task fib_task;
   fibonacci fib_n_1 (n-1), fib_n_2 (n-2);

    pfunc::spawn (fib_task, fib_n_1);
    fib_n_2();
    pfunc::wait (fib_task);

   answer=fib_n_1.answer+fib_n_2.answer;
  }
 }
 int answer;
 const int n;
};
```

Fibonacci (n=37)



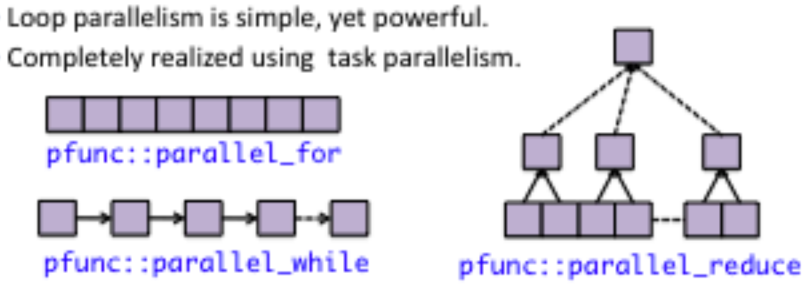
- 2× faster than TBB
- 2× slower than Cilk

PFunc-specific Features for Task Parallelism

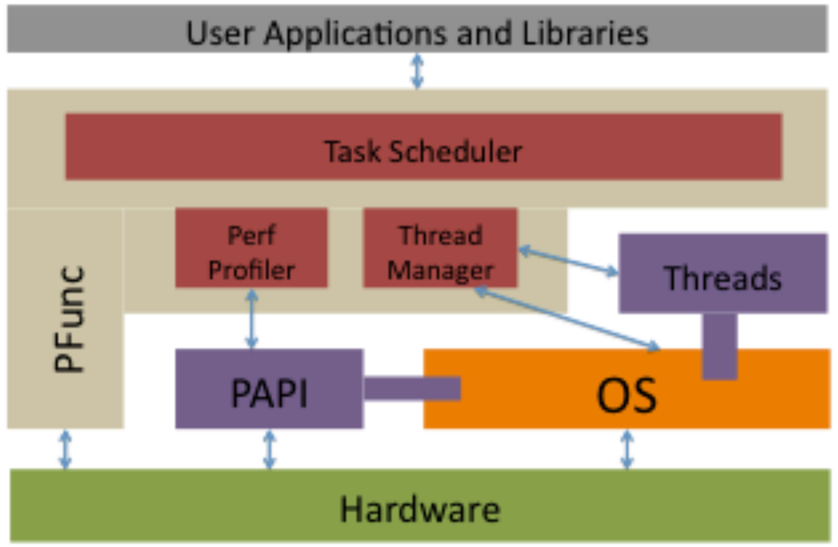
- Spawn tasks on specific queues.
- Select from work-stealing to work-sharing at runtime.
- Tasks can have multiple parents; execute DAGs seamlessly!

Loop Parallelism

- Loop parallelism is simple, yet powerful.
- Completely realized using task parallelism.



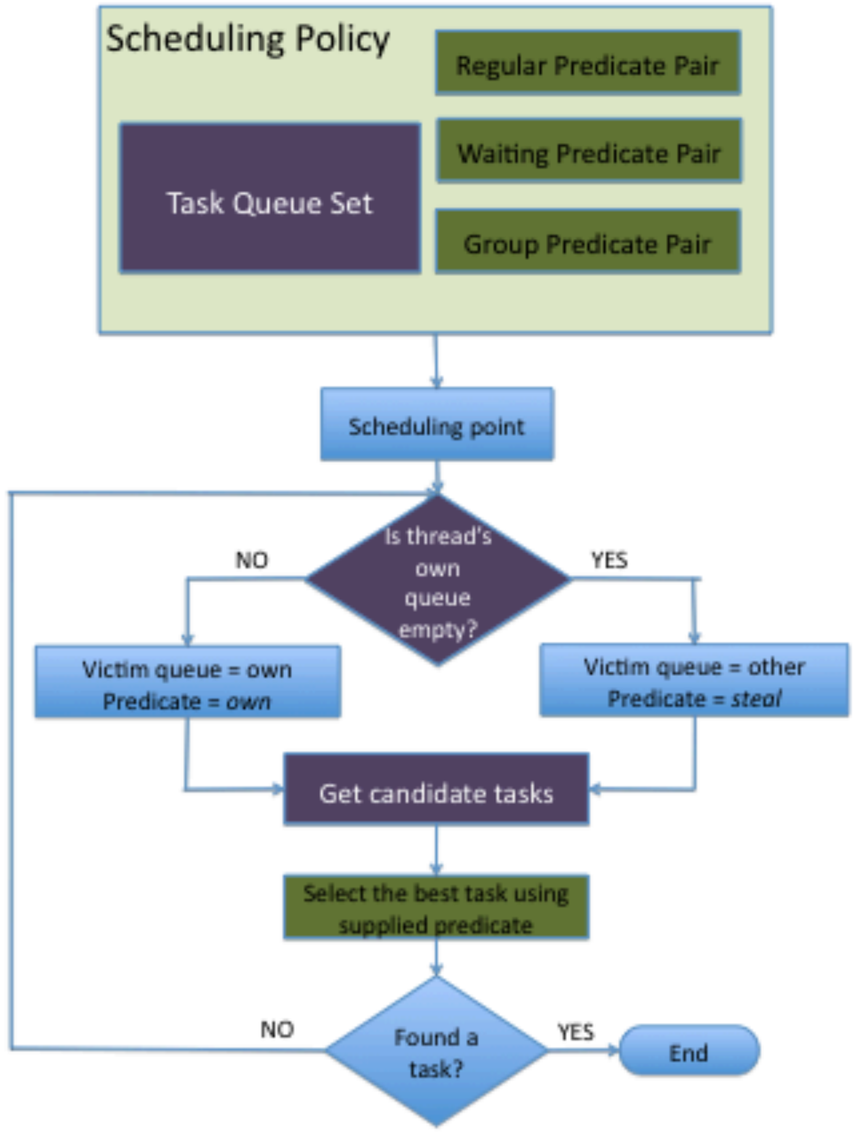
Software Architecture



Customizing at Compile-time

```
struct fibonacci;
typedef pfunc::generator <cilkS, /*Scheduling policy*/
                          pfunc::use_default, /*Compare*/
                          fibonacci> /*Functor*/ my_pfunc;

typedef my_pfunc::taskmgr taskmgr;
typedef my_pfunc::attribute attribute;
typedef my_pfunc::task task;
```



Feature	Built-in	Default
Scheduling policy	cilkS, prioS, fifoS, lifoS	cilkS
Compare	N/A	std::less<int>
Functor	N/A	virtual_functor

SPMD-style Parallelism

- Mix task parallelism with SPMD-style programming.
- Create groups of tasks; a task can be in only one group.
- Tasks can communicate/sync using their group rank.
- Barrier primitive on group allows collective syncs.

Pedagogical and Research Aids

- Portable, easy to install, and use.
- Thorough documentation and tutorials.
- Industry-strength exception handling.
- PAPI integration for profiling performance.
- Growing list of sample applications.
- Online user-groups and support.

Now Available: PFunc 1.0.

Operating System	Processor
Windows XP	x86_32
Linux	ppc32, ppc64, x86_32, x86_64
AIX	ppc32, ppc64
OS X	x86_32, x86_64

Salient Features

- New loop parallelism constructs.
- New examples including matmult, scale, and accumulate.
- Updated easy-to-use interface.
- Updated atomics: compare-and-swap, fetch-and-add, etc.

References

- https://projects.coin-or.org/PFunc
- PFunc: Modern Task Parallelism For Modern High Performance Computing, Kambadur et al., SC 2009.
- Demand-driven Execution Of Static Directed Acyclic Graphs Using Task Parallelism, Kambadur et al., HiPC, 2009.
- Extending Task Parallelism For Frequent Pattern Mining, Kambadur et al, ParCO, 2009.