PPP: Parallel Programming with Pictures

Annette Feng, Wu Feng, and Eli Tilevich
{afeng, wfeng, tilevich}@vt.edu

http://synergy.cs.vt.edu/

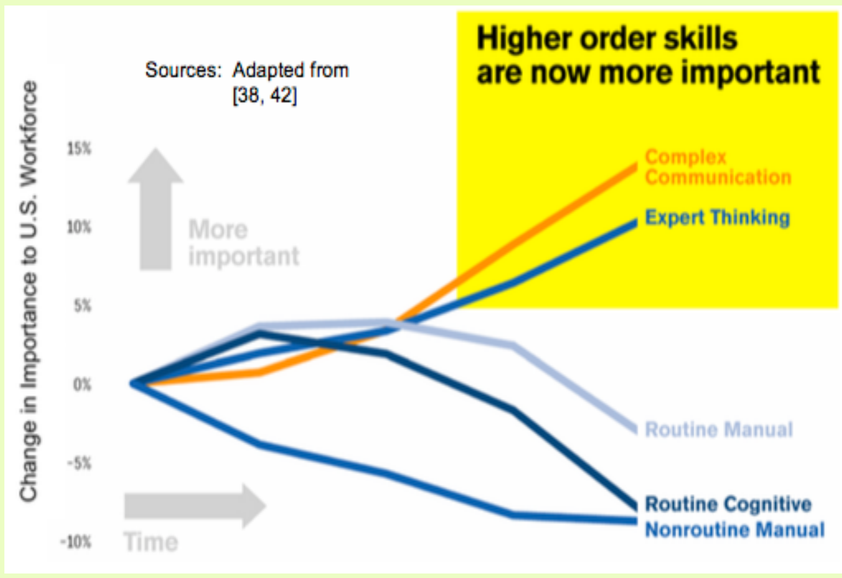
Motivation

Increasing Demand for Parallel Programmers ... from multi-core to supercomputing ...

1. Ubiquity of multi-core computing systems – supercomputers, desktops, laptops, embedded systems, and smartphones!
2. Lack of exposure to parallel programming by undergraduates
   - Parallel programming typically an *elective* class during the senior year and generally *not* taken because it's "too hard."
   - Undergraduates "brainwashed" with an ingrained sequential mindset in their approach to writing computer codes

Best Way to Create More Parallel Programmers?
Start them young! As young as middle school.

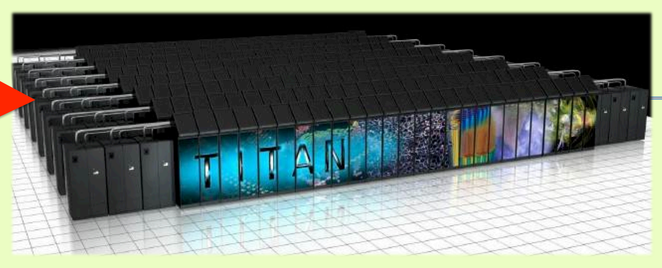
Technological Changes Affecting U.S. Workforce Skills



Approach

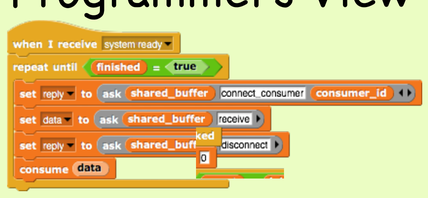
Research & develop building blocks for parallel programming

1. **Leverage** (sequential) picture-based programming environments, e.g., Scratch or Snap!
2. **Introduce** explicit building blocks for Parallel Programming with Pictures (PPP)
3. **Incorporate** appropriate infrastructure code for parallel execution in Scratch or Snap!
4. **Generate** back-end code so that parallel supercomputing systems can be supported, e.g., laptop → Tiny Titan at ORNL → Titan at ORNL.

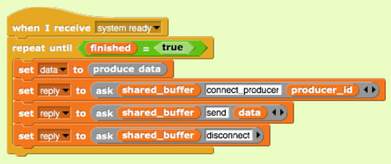


1. **Teach** parallel programming concepts
2. **Enable** explicit parallel programming in a blocks-based language
3. **Cater** to K-12 students, undergraduate students, or professionally re-training with easily understood abstractions
4. **Provide** general abstractions for coding multiple parallel paradigms
5. **Inform** by using the blocks in tutorials and example code

Programmer's View

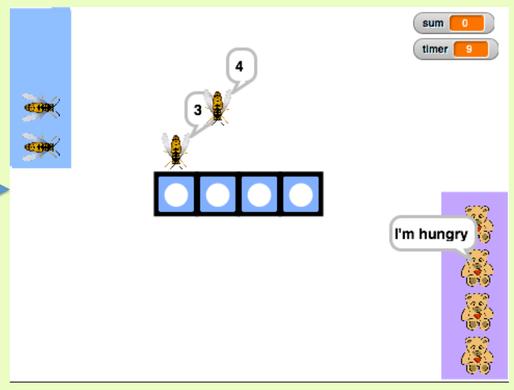


Producer Object



Consumer Object

Hidden Back-End Code



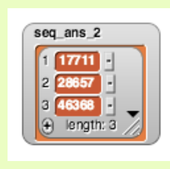
Output Environment for Visual Feedback

Hidden Back-End Code

```
Process.prototype.reportParallelFib = function(list) {
    var p;

    function fib(n) {if (n<2) {return 1} else {return fib(n-2) + fib(n-1)}};

    if (this.context.inputs.length < 2) {
        p = new Parallel(list.asArray());
        p.map(fib);
        this.context.inputs[1] = p; // store object in this.context.inputs...
    } else {
        p = this.context.inputs[1]; // ...to check in later runSteps for completion
        if (p.operation._resolved) {
            return new List(p.data);
        };
    }
    this.pushContext('doYield');
    this.pushContext();
};
```



Programmer's View

Conclusion

- Create a general and comprehensive set of parallel building blocks to enable explicit coding of multiple parallel paradigms

Future Work

- Expand the available parallel building blocks. (Current Support: Producer-consumer only)
- Generalize the back-end of "Parallel Programming with Pictures" (PPP) so that it can be adapted to other parallel supercomputing environments, including Tiny Titan and Titan at ORNL.
- Demo for hands-on interactivity at SC'15