What the Project is About

- SIMD hardware becoming increasingly popular
- Many applications are naturally recursive
- How can we map recursion to SIMD hardware?

Challenges

- HOW to put proper tasks together?
  - Easy to vectorize
- HOW to avoid putting too many tasks together?
  - Don’t use too many resources
- HOW to make sure we put enough tasks together
  - Avoid low SIMD utilization

High Level Idea of our Solution

Theoretical and Practical Evaluation Results

Theoretical Conclusion

For an n-node computation tree of depth \( d \), with a SIMD width of \( P \) and a block size of \( kP \), we have:

- For large block sizes \( (k > 2^{d - \lg n}) \), we can achieve asymptotically optimal parallelization.
- For medium block sizes \( (k \leq 2^{d - \lg n}, \text{but } k > 2^{d - \lg n}P) \) re-expansion is necessary to achieve optimal parallelism.
- For smaller block sizes, we cannot guarantee optimal parallelism (but we can achieve it in practice).

Future Work

- Generalize the kinds of applications we can handle:
  - New scheduling strategies to handle more complicated application structure
  - Data layout strategies to handle memory accesses
  - Unify task and data parallelism

- Generalize hardware platforms:
  - SIMD + multicore
  - GPUs

- Generalize theoretical results:
  - Stronger guarantees

Related Publication: 1. Bin Ren, Youngjoon Jo, Sriram Krishnamoorthy, Kunal Agrawal, Milind Kulkarni, Efficient Execution of Recursive Programs on Commodity Vector Hardware, The 36th annual ACM SIGPLAN conference on Programming Language Design and Implementation (PLDI), June, 2015. (To Appear)