SDH: MITCHELL

B. Gaynor[1], T. Giannakopoulos[1], P. Ilardi[1], R. Thompson[1], D. Kaeli[2], N. Agostini[2], K. Shivdikar[2], S. Jagannathan[3], Z. Zhou[3]

[1]Systems & Technology Research, [2]Northeastern University, [3]Purdue University

Specialized Functions: Software Defined Hardware (SDH)

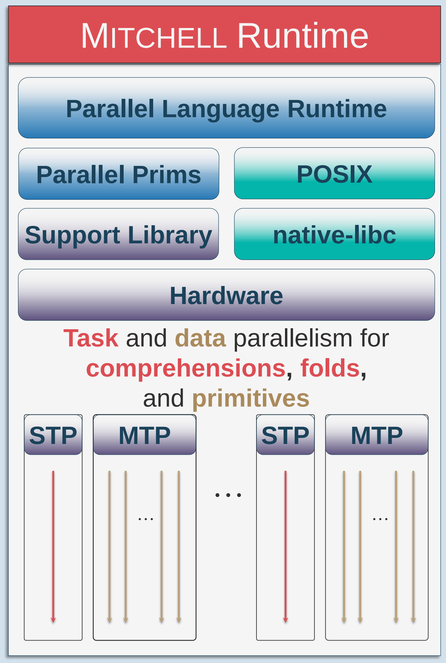
BACKGROUND

The **MITCHELL** programming language enables data scientists and machine learning practitioners to write performant machine learning and graph analytics programs that leverage highly-parallel hardware.

The MITCHELL compiler, runtime, and libraries provide language-managed parallelism, data-dependent run-time kernel selection, and safe approximation, thus enabling developers to specify algorithms easily while achieving high performance.

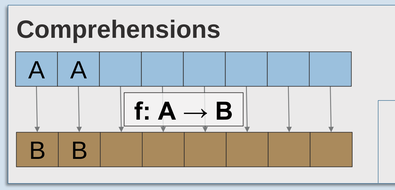
Automatic Parallelization And Optimization

APPROACH



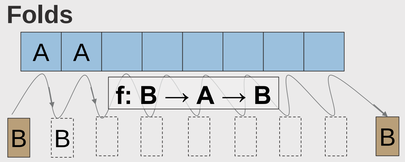
PHASE 1: Language-Managed Parallelism

- Task-parallelism via parallel execution of **comprehensions** and **folds**
- Data-parallelism via hardware-supported **primitive operations** on vectors, matrices, ...
- Libraries and abstractions to protect users from complexity
- **Accomplishment**: automated multi-threading for high-level programs on target hardware

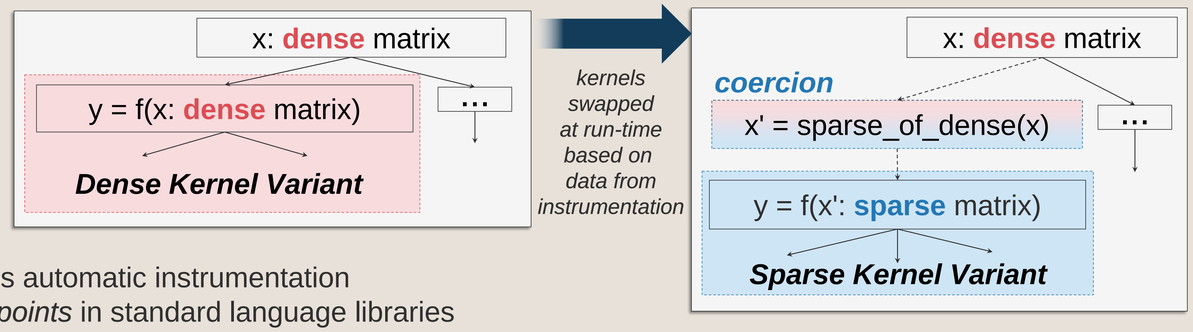


- No data dependency
- Minimal synchronization

- Loop carry dependency
- Tiling analysis



PHASE 2: Data-Dependent Run-time Kernel Selection



- Whole-program compilation enables automatic instrumentation based on annotated kernel *choice-points* in standard language libraries
- Data statistics collected and used to automatically select kernels and insert representation coercions at run-time

PHASE 3: Safe Approximation

- Getting good answers fast is often better than calculating perfect answers too late
- Built-in compile-time static analysis and run-time instrumentation determine when it is safe to use approximate kernels
- Builds on Phase 2 automatic kernel selection mechanism to realize additional algorithm accelerations

IMPACT: Phase 1

High-Level Programming Targets Bare-Metal Hardware With Massive Parallelism

- Parallelism without explicit concurrency
  - Programmers take advantage of parallel hardware without explicit thread and synchronization management
- Hardware support for performant language primitives

Calculating the Update for Gradient-Boosted Decision Trees

⇉ Parallel Comprehension , ⇆ Parallel Fold , Parallel Primitive

```
1.  fun calculateUpdate gbdt data learningRate =
2.    let
3.      fun loss (features, label) = lossR gbdt (features, label)
4.  ⇉   val dataGradient = List.map loss data
5.      val dt = CART.train dataGradient
6.      val family = CART.prune dt data
7.  ⇆   val avg = averageOf_i (fn (_, dt) => DT.leafNum dt) family
8.      fun comp left right = compareLeaves avg left right
9.  ⇆   val best = argmax comp family
10.     val dt = case best of NONE => dt | SOME (_, x) => x
11.   in
12.     applyLearningRate dt learningRate
13.   end
```

Multiple Integrated Threading Models

See Mitchell Runtime diagram to far left

- Comprehensions and folds provide task-parallelism
  - Run on single-threaded pipelines (STPs)
- Primitive operations provide data-parallelism
  - Run on multi-threaded pipelines (MTPs)

Efficient Primitive Operations

- Specialized to SDH hardware
- Libraries for matrices, vectors, graphs, decision trees, ...

BLAS Level	Function	Description
1	void cblas_<>scal	Multiply array by scalar
1	void cblas_<>copy	Copy array to array
1	void cblas_<>axpy	Add arrays element-wise
1	<> cblas_<>dot	Array dot product
1	float cblas_sdsdot	Array dot product plus constant
2	void cblas_<>gemv	Generic matrix-vector multiplication
2	void cblas_<>spmv	Sparse matrix-vector multiplication
⋮	⋮	⋮

BLAS & GSL Primitives With Hardware Support

CONTACT

Brad Gaynor
Brad.Gaynor@STResearch.com

SYSTEMS & TECHNOLOGY RESEARCH

This research was developed with funding from the Air Force Research Laboratory (AFRL) and the Defense Advanced Research Projects Agency (DARPA). The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government.

THE ELECTRONICS RESURGENCE INITIATIVE

2019-05-03