CIFellows 2020-2021

Computing Innovation Fellows

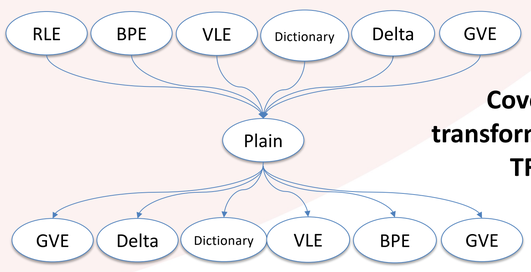
Portable Programming of High-performance Data Transformation

Marziyeh Nourian — University of Chicago
Mentor: Dr. Andrew A. Chien

Introduction and Objectives

- Explosive rise of "big data" and data-intensive computing calls for efficient data representations. Efficient data representations allow for storage, movement, and computational efficiency.
- Conventional sequential processors and programming models are not designed for efficient data transformation. Transformation of different data representations is a key performance challenge, limiting the use of representations with expensive transformations in practice.
- We propose a **computational model called extended Deterministic Finite-state Transducer (DFST+)** and a **high-level programming model called Transducer Form (TFORM)** that allows for a **compact, portable and efficient implementation** of data transformation

Parquet (a data storage format) Encodings



Covering NxN custom transformations with only 2xN TFORM programs

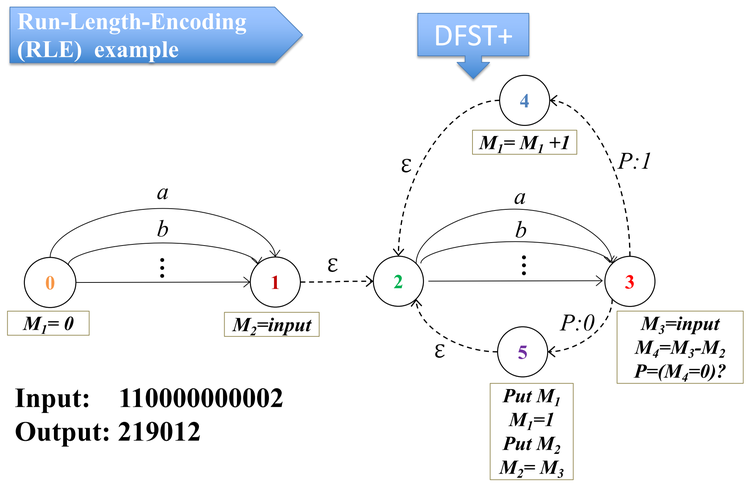
- We use TFORM programs for efficient data transformation on CPU, Unstructured Data Processor (UDP) [1] (a general data transformation accelerator), and UpDown Engine [2] (a memory-movement and recode accelerator embedded in memory hierarchy)

Data Transformation Computational and Programming Model

- Extended Deterministic Finite-State Transducer (DFST+) extends DFST (traditional computational model for data transformation) with variables, actions on the variables, and transitions conditional to the variables to enable compact and efficient representation of data transformation
- TFORM programming model enables portable expression of DFST+

Run-Length-Encoding (RLE) example

DFST+

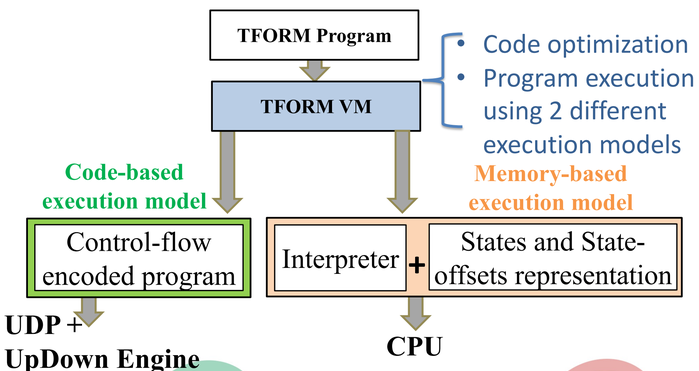


Input: 110000000002
Output: 219012

TFORM Program

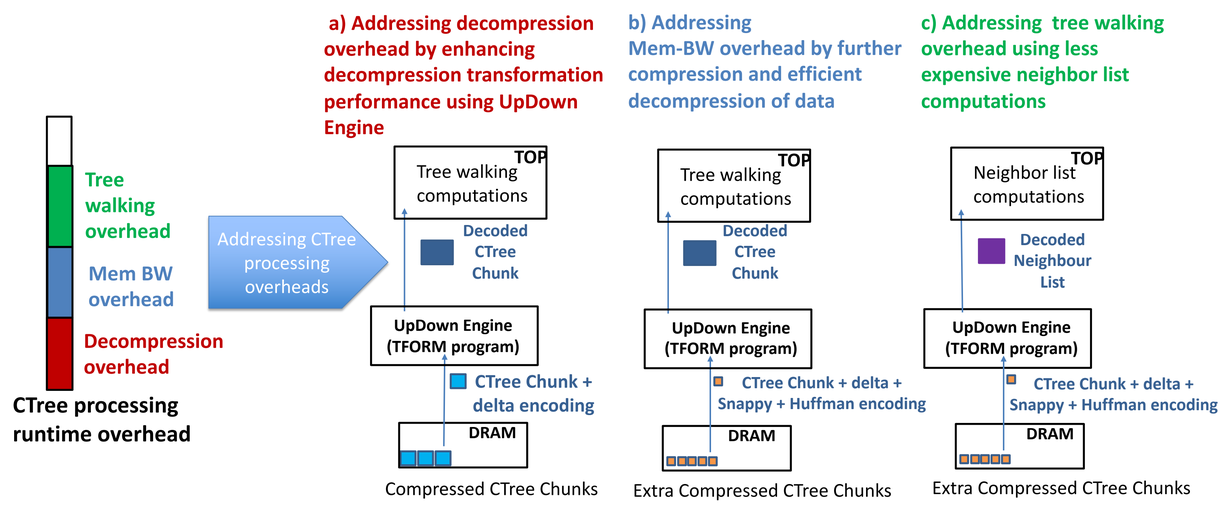
```
BLOCK 0{
      M[1]=0
      if(INPUT=a) GOTO BLOCK 1
      …
}
BLOCK 1{
      M[2]= INPUT
      GOTO BLOCK 2
}
BLOCK 2{
      if(INPUT=a) GOTO BLOCK 1
      …
}

BLOCK 3{
      M[3]= INPUT
      M[4]=M[3]-M[2]
      PREDICATE= M[4]==0
      GOTO BLOCK 5 IF PREDICATE == 0
      GOTO BLOCK 4 IF PREDICATE == 1
}
BLOCK 4{
      M[1]=M[1]+1
      GOTO BLOCK 2
}
BLOCK 5{
      put M[1]
      M[1]=1
      put M[2]
      M[2]=M[3]
      GOTO BLOCK 2
}
```

- TFORM Virtual Machine (VM) allows for reducing DFST+ to practice



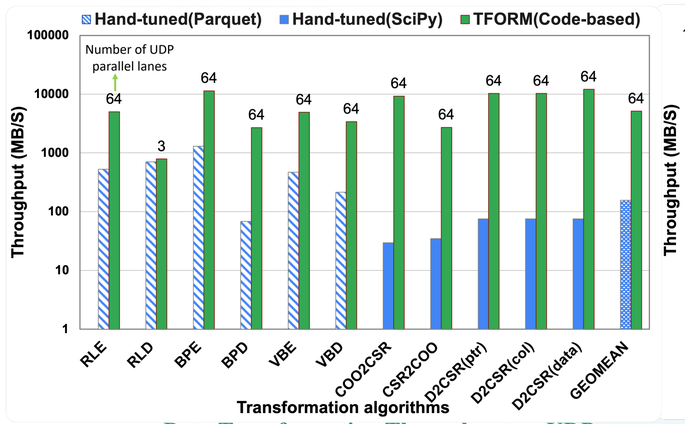
Applications

- Data analytics systems: Enhancing performance of transformations exploited in Parquet (data format commonly used in data analytics systems) library
- Sparse matrix computations: Enhancing performance of transformations for different sparse representations and encodings
- Graph processing: Enhancing performance of using compressed functional tree (CTree) representation of graphs [3] (ongoing work)

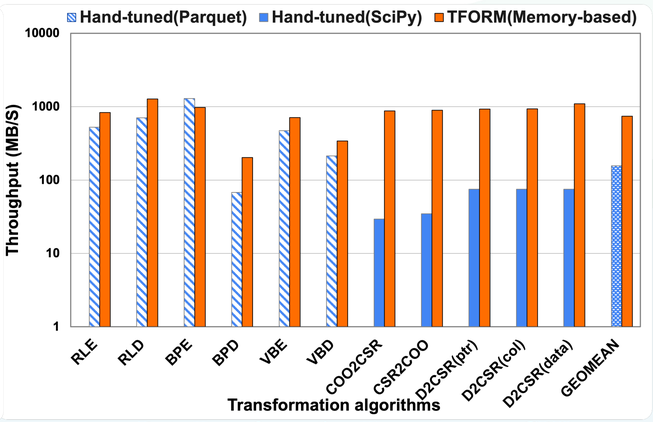


Experimental Evaluation

- We compare performance of Parquet and SciPy (for sparse matric transformation) library on CPU (64 cores) with CPU and UDP implementation of TFORM-based transformations



Data Transformation Throughput on UDP
17.8X speedup (geomean)



Data Transformation Throughput on CPU
1.8X speedup over Parquet and 18.9X speedup over SciPy (geomean)

Conclusion

- We propose DFST+ (a new data transformation model), TFORM (a data transformation programming model that expresses DFST+), and TFORM VM to efficiently implement TFORM programs, enabling superior performance gains on UDP accelerator and competitive performance on CPU compared to the hand-tuned libraries
- We exploit portable and efficient TFORM programs to unlock the power of existing and future data representations and hardware accelerators for efficient data transformation.

References

[1] Y. Fang, C. Zou, A. J. Elmore, and A. A. Chien. "UDP: A programmable accelerator for extract-transform-load workloads and more.", In Proceedings of MICRO 2017.

[2] A. A. Chien, A. Rajasukumar, M. Nourian, *C. Zho, and Y. Fang,* "Updown Instruction Set Architecture v0.9", Dept. Computer Science, University of Chicago, Tech. Rep., TR-2022-02, 2022.

[3] L. Dhulipala, G. E. Blelloch, and Julian Shun. " Low-latency graph streaming using compressed purely-functional trees.", In Proceedings of PLDI 2019.

CRA Computing Research Association
CCC Computing Community Consortium Catalyst
NSF