DryadLINQ

A System for General-Purpose Distributed Data-Parallel Computing Using a High-Level Language
Overview

- Motivation for DryadLINQ
- Design
- Implementation
- Performance
- Q & A
Motivation

- More machines + more code = more problems
- Need to simplify!
- **Solution → Higher-level Language**
Design Goals

- Easy to write
- General Purpose
- Efficient
Existing Solutions

- **SQL**
  - Difficult to express common programming constructs

- **MapReduce**
  - Not flexible enough
  - Inefficient for some use cases

- **Dryad**
  - Have to specify DAG
  - Harder to write
DryadLINQ

- **Dryad**
  - Execution Engine
- **Language INtegrated Query**
  - Declarative + Imperative + Object Oriented
LINQ vs. SQL

- Expressions can be directly embedded in code
- Allow direct calls to C#, F#, ... functions
- Evaluated by Dryad
• Declarative

```csharp
var adjustedScoreTriples =
    from d in scoreTriples
    join r in staticRank on d.docID equals r.key
    select new QueryScoreDocIDTriple(d, r);
```

• OO

```csharp
var adjustedScoreTriples =
    scoreTriples.Join(staticRank,
    d => d.docID, r => r.key,
    (d, r) => new QueryScoreDocIDTriple(d, r));
```
API

- Compatible with many .NET Languages (e.g. C#)
- DryadLINQ vs. SPARK
  - Language embedded
  - Compiler Hints
  - Functions must have no side effects
  - Non-interactive
Data Model

- `IEnumerable<T>` vs. RDD’s
  - Distributed
  - Strongly typed
  - Mutable
  - Nested generics
  - Lazy Evaluation
Execution

- Similar to SQL query plan
- Create execution plan graph
- Static Optimizations
- Pass to Dryad Job Manager
- Dynamic Optimizations
Expression Execution

// Do Stuff ...
var DT = ToDryadTable(X);
foreach (row in DT) {
    // Do more stuff ...
}

Figure 2: LINQ-expression execution in DryadLINQ.
Optimizations

- Static
  - I/O reduction
  - Pipelining
  - Eager aggregation

- Dynamic
  - Partitioning
  - Topology aware aggregation
  - Lazy evaluation
Example: OrderBy
Performance

- TeraSort
- Skyserver Q18 computation
TeraSort

\[ \approx 3.87 \text{ Gb per machine} \]
Comparison

![Comparison Graph](image)

- Dryad Two-pass
- DryadLINQ

**Legend:**
- Blue line: Dryad Two-pass
- Red line: DryadLINQ

**Axes:**
- Y-axis: Speed-up
- X-axis: Number of computers