SPADE: the System S declarative stream processing engine.

by Gedik, Bugra, et al from IBM, University of Illinois and Georgia Tech
What is SPADE?

- A declarative stream processing engine and language developed at IBM.
- Compiles and optimises stream-based SPADE code.
- Programs are deployed by IBM’s System S.
- One of the early birds in “program like a stream” stream processing (2008).
  - Spark (2010), Storm (2011)
What is System S?

- A large-scale distributed data stream processing middleware
- Takes jobs, e.g. Data-Flow Graphs, a set of processing elements (PEs) and then distributes these on the cluster
- Comes as a C++ library and a high level inquiry engine. “Estimate customer satisfaction”
- Used for fault-tolerance, deployment, scheduling etc for SPADE
SPADE’s programming model

- Program in terms of streaming operators
  - SPADE supplies common relation algebra ones
  - Developers can add their own

- Operators compiled into Processing Elements (PEs)

- External inputs compiled as sources, outputs as sinks
SPADE source code
System S

Source
Twitter stream

Source
BBC RSS

Source
Twitter stream

Source
BBC RSS

Sink
US DB

Sink
European DB

Sink
US DB

Sink
European DB

SPADE source code

PE
<User func>

PE
Sort
Split

PE
Aggregate

T1: Sort
T2: Split

PE
Aggregate

PE
<User func>
All PEs can be put into a single system too!
Multiple operators in one PE

- Minimise communication overhead.
- Ensures two or more operators are scheduled on the same machine.
- Automatic thread safety even for user-defined operators
SPADE compilation

- Compilation specific to underlying system and network topology for better performance

- Support for different types of windows
  - Sliding windows, punctuations, tumbling

- Compile in a special mode for statistics collection to analyse the properties of the program.

- Recompilation then possible to optimise further due to known heuristics
Incremental Deployment and Programs

- SPADE supports live updates to running programs.
- Like other streaming frameworks it is suitable for incremental algorithms.
SPADE source code

New source

SPADE

System S

Source
Twitter stream

Source
BBC RSS

PE
Aggregate

PE
T1: Sort  T2: Split

PE
<User func>

Sink
US DB

Sink
European DB
SPADE source code

New source

SPADE

PE
<User func2>

System S

Source
Twitter stream

Source
BBC RSS

PE
Aggregate

PE
<T1: Sort
T2: Split>

Sink
US DB

Sink
European DB
SPADE source code

New source

SPADE

System S

Source
- Twitter stream
- BBC RSS

Sink
- US DB
- European DB

Source
- <User func2>

Sink
- <User func>

Sink
- <User func2>

T1: Sort
T2: Split
Development IDE
Results

Figure 6: Tuple ingestion rate for the parallel and distributed bargain index computation application, using 22 parallel queries distributed over 16 nodes.
Strengths 1/2

- Declarative topology (through operators)
- Extensible operators
- Good performance ratio to programming difficulty
Strengths 2/2

- Intermediate results made available
- Incremental algorithms / deployment
- Natural development environment
- Highly influential for newer systems
Criticisms 1/2

- Uses raw numbers in results without context or comparison.
- Only one arbitrary experiment carried out.
- Fixed number of nodes for test
Criticisms 2/2

- How well does thread-locking with user defined operators work in practise?

- Long compilation times and system specific compilations.
Questions?