Ziemlich langer Titel der studentischen Arbeit

Bachelor-Arbeit
Niels Danger
KOM-type-number

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Fachbereich Elektrotechnik
und Informationstechnik
Fachbereich Informatik (Zweitmitglied)

Fachgebiet Multimedia Kommunikation
Prof. Dr.-Ing. Ralf Steinmetz

Ziemlich langer Titel der studentischen Arbeit
Bachelor-Arbeit
KOM-type-number

Eingereicht von Niels Danger
Tag der Einreichung: 18. September 2017

Gutachter: Prof. Dr.-Ing. Ralf Steinmetz
Betreuer: Manisha Luthra
Externer Betreuer: Dr. Nokia Siemens

Technische Universität Darmstadt
Fachbereich Elektrotechnik und Informationstechnik
Fachbereich Informatik (Zweitmitglied)

Fachgebiet Multimedia Kommunikation (KOM)
Prof. Dr.-Ing. Ralf Steinmetz

Ehrenwörtliche Erklärung

Hiermit versichere ich, die vorliegende Bachelor-Arbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in dieser oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen. Die schriftliche Fassung stimmt mit der elektronischen Fassung überein.

Darmstadt, den 18. September 2017 Niels Danger

Contents

Abstract

1 Introduction

1.1 Problem Description

With the ever increasing number of hard- and software systems that generate continuous streams of heterogeneous data it becomes more difficult to extract meaningful information from this data in real time. Every data item that is generated can be viewed as an event, in the sense that two identical data items created at different points in time are two distinct events. This is the application domain of Complex Event Processing (CEP): Consumers can define queries as complex events that represent situations (or aspects of them) and event patterns that are of interest to them. Producers generate streams of data in the form of simple events (e.g. sensor readings) which can be aggregated and combined with other events (simple or complex) and domain knowledge to create complex events. If relevant events are detected in the event stream, the CEP engine will notify the interested consumer(s) immediately. This allows for timely reaction to the new situation by the consumers.

If the processing of the events happens in a centralized fashion, scaling of the system (in terms of joining producers, consumers or an increase of the event rate) is obviously limited. Therefore, distributing the processing of the data over several computing nodes is often beneficial, if not necessary. This becomes even more relevant if the producers and consumers are geographically spread out and short latencies from producers and consumers are important. The general problem can be described as follows: the query, in the form of a logical flow graph consisting of sources, operators and drains is to be mapped onto an underlying network of processing nodes that is connected to the producers and consumers. The sources and drains are pinned to the producers and consumers, so the operators are the only moveable part of the graph. This gives rise to the question as to how to optimally place the individual operators involved in satisfying the consumers' queries. The notion of what an optimal placement is can vary substantially depending on the application scenario and its specific Quality of Service (QoS) requirements. These QoS metrics can be, among others:

- latency
- bandwidth utilization
- load balance
- cost of transport and processing
- reliability
- availability
- energy consumption

The deployment of the operators to the nodes has to happen in such a manner that the most important QoS metric(s) are optimized. However, the placement of the operators cannot happen in an arbitrary way: some operators require a minimum of processing power or specific hardware, so this can reduce the number of feasible solutions. This is further constrained by sequential dependencies between operators: if an operator combines several simple or complex events, the operators that process these events need to be placed on processing nodes located upstream (viewed from the drain) of this operator's processing node.

Currently, most operator placement algorithms focus on optimizing latency and/or load balancing. While most of them are capable of reacting to changes in the CEP system and the underlying network to a certain degree, there is currently no way to adapt the utilized placement algorithm to major changes of the application environment that affect the QoS requirements. If the consumers' preferences of QoS metrics is changed, the placement algorithm that was used before may no longer be optimal as it was designed with a different goal in mind. Such a change of preferences can be caused by a change of

application context and could be detected by evaluating contextual information. This makes the ability to adapt the deployed operator placement algorithm beneficial for reacting to changes in the desired properties of the system.

1.2 Goals

The goals of this thesis are as follows:

- identify and assess QoS metrics w.r.t. their suitability and relevance for application scenarios
- design and implement a heuristic that optimizes operator placement for a given set of QoS metrics

2 Paper Summary

The paper Placement Strategies for Internet-Scale Data Stream Systems by Geetika T. Lakshmanan, Ying Li and Rob Strom is a survey of 8 algorithms developed to solve the problem of optimal operator placement in data stream processing systems. To compare the different approaches, the authors define a set of core components that significantly affect the performance of the system and compare the categories of these components:

- architecture (centralized, decentralized or hybrid implementation of placement logic)
- algorithm structure (centralized vs decentralized decisions)
- metric(s) to optimize (load, latency, bandwidth, machine resources, operator importance, combinations of those)
- operator-level operations (reuse or replication of operators)
- reconfiguration (triggers for operator migration: thresholds, periodic re-evaluation, constraint violation)

Based on these components, 8 placement algorithms developed respectively by Pietzuch, Balazinska, Abadi, Zhou, Kumar, Ahmad, Amini and Pandit are assigned to the discussed categories. The authors then proceed by comparing the algorithms w.r.t. their design decisions and underlying assumptions. It is concluded that decentralized approaches which are able to adapt dynamically by operator migration are prevalent, with latency and, by extension, load balancing being the preferred metrics to optimize. Based on the comparison of the assumptions a decision tree for selecting a fitting placement algorithm considering the characteristics of a given problem (node distribution, number of administrative domains, dynamics of topology and data, query diversity) is proposed.