ISSN 1598-0170 (Print)
ISSN 2287-1136 (Online)
http://www.jksii.or.kr

스트림-리즈닝을 위한 실시간 사물인터넷 빅-데이터 처리☆

Real-Time IoT Big-data Processing for Stream Reasoning

윤 창 호[1,2] 박 종 원[1,2] 정 혜 선[1,2] 이 용 우[1,2*]
Chang Ho Yun Jong Won Park Hae Sun Jung Yong Woo Lee

요 약

스마트-시티는 스마트-시티의 사물인터넷(Internet of Things: IoT) 디바이스를 비롯한 수많은 인프라를 지능적으로 관리하고, 다양한 스마트 어플리케이션을 도시민에게 제공한다. 스마트-시티에서는 스마트-시티 어플리케이션에서 필요한 다양한 정보를 제공하기 위하여 수많은 사물인터넷 기기들로부터 끊임없이 발생하는 대규모의 스트림 빅-데이터를 지능적으로 처리하는 기능이 필요하다. 하지만, 스마트-시티에서 대규모의 스트림 빅-데이터를 처리하는 것에는 실시간 처리와 관련된 제약들이 존재한다. 본 스마트-시티-사업단에서는 선행 연구에서 스마트-시티미들웨어와 이를 이용한 스트림-리즈닝 방법론 및 시스템을 개발하였다. 스마트-시티에서 스마트 서비스를 제공하기 위하여, 스마트-시티-사업단에서는 스트림-리즈닝을 사용하는 방법론을 사용한다. 이 스트림-리즈닝은 대용량 데이터의 실시간 처리를 필요로 한다. 따라서, 후속연구로서 스마트-시티미들웨어의 클라우드-컴퓨팅 플랫폼을 이용하여 스트림-리즈닝을 위한 실시간 분산병렬처리 클라우드-컴퓨팅 방법론과 시스템을 개발하였다. 본 논문에서는 스마트-시티에서 발생하는 사물인터넷 빅-데이터를 스트림-리즈닝에 사용하기 위하여 이 후속연구에서 개발된 클라우드 기반 실시간 분산병렬처리 연구결과를 소개한다. 스마트-시티의 각종 센서들로부터 전송되어지는 사물인터넷 빅-데이터를 사용하여 스트림-리즈닝하는 데 필요한 클라우드-컴퓨팅 기반의 실시간 분산처리 방법론과 시스템을 소개하고 있으며, 이 방법론을 선행연구에서 개발한 스마트-시티 미들웨어에 구현하여 실시간 분산처리 성능을 평가한 것을 소개한다.

☞ 주제어 : 스마트-시티, 미들웨어, 스트림리즈닝, 실시간 처리, 클라우드-컴퓨팅.

ABSTRACT

Smart Cities intelligently manage numerous infrastructures, including Smart-City IoT devices, and provide a variety of smart-city applications to citizen. In order to provide various information needed for smart-city applications, Smart Cities require a function to intelligently process large-scale streamed big data that are constantly generated from a large number of IoT devices. To provide smart services in Smart-City, the Smart-City Consortium uses stream reasoning. Our stream reasoning requires real-time processing of big data. However, there are limitations associated with real-time processing of large-scale streamed big data in Smart Cities. In this paper, we introduce one of our researches on cloud computing based real-time distributed-parallel-processing to be used in stream-reasoning of IoT big data in Smart Cities. The Smart-City Consortium introduced its previously developed smart-city middleware. In the research for this paper, we made cloud computing based real-time distributed-parallel-processing available in the cloud computing platform of the smart-city middleware developed in the previous research, so that we can perform real-time distributed-parallel-processing with them. This paper introduces a real-time distributed-parallel-processing method and system for stream reasoning with IoT big data transmitted from various sensors of Smart Cities and evaluate the performance of real-time distributed-parallel-processing of the system where the method is implemented.

☞ keyword : Smart-city, Middleware, Stream reasoning, Real-time processing, cloud computing.

1. 서 론

스마트-시티(Smart-City)는 도시민의 삶의 질을 향상시키기 위해서 스마트 서비스들을 언제, 어디서나, 어느 때, 어떠한 디바이스를 통해서도 도시민에게 제공하기 위해

1 School of Electrical and Computer Engineering, University of Seoul, 163 Seoulsiripdaero, Dongdaemun-gu, Seoul, 02504, Korea.
2 Smart City Consortium, 71-7, Nogyang-ro, Uijeongbu-si, Gyeonggi-do, 11614, Korea.
* Corresponding author (Yong Woo Lee: ywlee@uos.ac.kr)
[Received 4 November 2016, Reviewed 22 November 2016(R2 3 March 2017), Accepted 17 April 2017]
☆ This work was supported by the 2014 Research Fund of the University of Seoul. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2015R1C1A1A02036461). Part of this study was supported by the Seoul Research and Business Development Program, Smart City Consortium (10561) and Seoul Grid Center.

서 도시에 정보통신기술이 녹아져 있는 미래형 도시이다. [1] 사물인터넷기술은 실세계에 존재하는 사물 (Physical Things) 및 사이버 환경에 존재하는 사물 (Virtual Things) 들이 인터넷을 통하여 서로 연결되고 이러한 물리공간과 가상공간의 사물, 데이터, 사람들이 연동을 통하여 다양한 서비스를 제공할 수 있는 미래 인터넷 인프라 기술이다.

스마트-시티에서 스마트 서비스를 구현하기 위해서는 사물인터넷의 스마트 기기로부터 전송되는 엄청난 양의 빅-데이터를 분석하여 정확한 상황인식을 통한 판단과정을 거쳐서, 도시민에게 지능적인 서비스를 제공하는 것이 필요하다. 이때, 끊임없이 발생하는 빅-데이터를 실시간으로 처리하여 상황인식을 하는데 스트림-리즈닝(Stream Reasoning) 기술이 매우 유용하다. 스트림-리즈닝은 최근에 발생한 분야로서, “매우 많은 수의 동시 사용자들이 사용하는 의사 결정 시스템을 지원하기 위한 목적으로, 다수의 이질적이며 불량 데이터가 섞인 거대한 규모의 빅-데이터 스트림을 실시간으로 처리하는 논리적 추론 (Logical Reasoning)” 을 말한다. [2]

본 스마트-시티-사업단은 선행 연구에서 스마트-시티를 효율적으로 관리하기 위해서 3티어 패러다임인 유토피아 (UTOPIA)를 개발하였다. 유토피아는 스마트-시티 인프라스트럭처 티어, 스마트-시티 미들웨어 티어, 스마트-시티 포탈티어로 구성되어 있으며, 이 중 가장 핵심이 되는 스마트-시티 미들웨어 티어는 사람의 뇌, 즉 영혼과 같은 역할을 하기 때문에, 소울(SOUL: Smart and Outstanding Ubiquitous Lounge)이라고 이름 지었다.

소울은 스마트 서비스를 제공하기 위하여 스트림-리즈닝 기술을 사용한다. 하지만, 이 스트림-리즈닝에서 사용하는 스트림 데이터는 빅-데이터로서, 빠른 속도로 변하면서 끊임없이 계속해서 공급되기에, 실시간으로 처리해야만 하는 매우 해결하기 어려운 제약조건을 소울이 해결하도록 요구하고 있다. 본 논문에서는 이 문제를 해결한 결과를 소개한다.

본 연구의 기여효과는 아래와 같다. 1) 클라우드-컴퓨팅 기술을 스트림-리즈닝에 융합하여 실시간 스트림-리즈닝을 할 수 있는 방법론을 개발하고 이를 소울에 구현하였다. 이렇게 함으로서, 소울은 사물인터넷으로 부터 끊임없이 연속적으로 빠르게 변화하면서 소울로 전송되는, 빅-데이터를 실시간으로 스트림-리즈닝할 수 있게 되었다. 따라서, 미들웨어와 스트림-리즈닝, 클라우드-컴퓨팅을 융합하여 스마트-시티에서 실시간으로 스트림-리즈닝을 할 수 있게 한 것이 본 연구와 논문의 기여효과이다.

2) 개발된 시스템의 성능을 평가하여 개발된 방법론과 시스템의 작동을 확인하고 주어진 부하에 적합하면서 확장성있는 시스템 환경을 알아내었다. 따라서, 스마트-시티에서 최적의 미들웨어를 이용한 실시간 스트림-리즈닝 시스템을 구축한 것이 본 연구와 논문의 기여효과이다.

본 논문의 구성은 다음과 같다. 1장에서 서론을 기술하였다. 이어 2장의 관련연구에서는 사물인터넷 과 스트림-리즈닝 연구에 관련된 것 중에서 본 논문의 주제와 가깝다고 판단되는 연구들에 대해 기술한다. 3장에서는 본 논문의 선행 연구에서 개발된 스마트-시티 패러다임과 스마트-시티를 위한 미들웨어에 대하여 설명한다. 4장에서는 스마트-시티 시스템과 스트림-리즈닝을 기반으로 하여 클라우드-컴퓨팅과 실시간 처리 기술들을 융합함으로서, 실시간 스트림-리즈닝이 가능하게 하는 방법론과 시스템에 대하여 상세히 서술한다. 5장에서는 구현한 시스템의 성능을 평가한다. 마지막으로 6장의 결론을 끝으로 본 논문을 마무리한다.

2. 관련연구

수많은 사물인터넷 기기들로부터 끊임없이 입력되는 빅-데이터를 처리하는 스트림-리즈닝 연구는 현재까지 찾아보기 어려웠다. 그러나, 최근에, 빅-데이터를 처리하는데 사용할 수 있어 보이는 오픈소스 소프트웨어가 발표되고 있다. 이 중에서, 본 연구진은 실시간-처리에 사용가능하다고 판단되어지는 오픈-소스-소프트웨어로서 아파치 스톰(Apache Storm) [3], 아파치(Apache) S4 [4], 아파치 삼자(Apache Samza) [5] 등이 유용하다고 판단했다. 마이크로 배치 프로세싱 방법을 사용하는 오픈-소스-소프트웨어로는 아파치 스파크 (Apache Spark) [6]를 들 수 있다.

[7]은 클러스터 컴퓨팅 시스템과 야후(Yahoo) S4 프레임워크를 활용한 병렬 처리 방법론을 제시하고 있다. 또한 이 방법론에서, 하위수준 술어와 이의 집합을 제시하고 이들을 이용하여 RDFS 추론의 인코딩 하는 방법과 C-SPARQL 질의 응답 하는 방법을 설명하고 있다.

[8]는 대규모 실시간 시맨틱 처리 프레임워크와 사물인터넷 어플리케이션을 위한 탄력적인 분산 스트림 엔진을 제안하고 있다. 제안된 엔진은 스파크를 사용하고 있으며, 사물인터넷 어플리케이션을 지원한다. 이 논문은 거주 환경을 모니터링하는 어플리케이션의 사례를 소개하고 있다.

[9]는 사물인터넷 데이터에 최신 시맨틱 기술을 적용하여 개발된 시맨틱 추론 시스템을 소개하고 있다. 다양한 시맨틱 데이터 포맷이 사용가능하며, 단일 추론기, 분

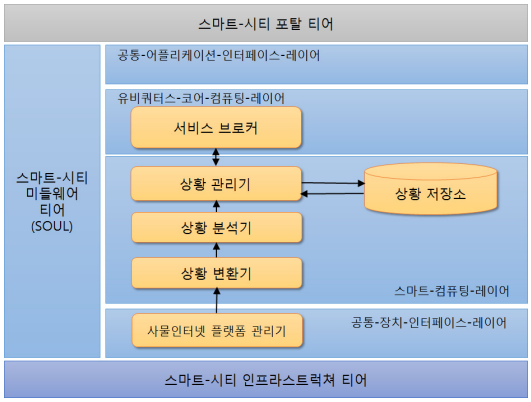
산 추론기, 모바일 추론기와 이들을 통합한 통합 추론기를 지원한다. 이 들은, 서로 다른 추론 접근법을 지원하는 확장성의 특성을 가지고 있다. 이 논문은 실시간 응답 시간을 비교하여, 사물인터넷 어플리케이션을 위한 시맨틱 기술의 성능을 평가하였다. 또한 추론처리에 사용할 수 있는, 분산된 사물인터넷 데이터 통합 전략을 비교 평가하였다.

위 연구들은 클라우드-컴퓨팅 기술을 이용한 점과 스트림-리즈닝의 처리 속도를 증가시키고자 한 점에서는 본 연구와 연관성이 있다. 그러나, 우리의 연구와는 달리, 미들웨어를 기반으로 하고 있지 않기 때문에, 사용의 편의성, 확장성 등에 있어서 제한이 있다. 우리의 연구는 스마트-시티 미들웨어를 기반으로 하기 때문에, 많은 장점과 차이점을 가지고 있다. 즉, 우리의 방법론과 시스템에서는, 이질적인 사물인터넷 기기로 부터의 스트림 빅-데이터를 실시간으로 통합 처리할 수 있으며, 다양한 실시간 클라우드-컴퓨팅 기법을 어플리케이션의 특성에 따라 선택적으로 적용시킬 수 있다. 또한 우리의 연구결과는 데이터 전송의 신뢰성과 확장성을 보장하는 면에서 타 연구결과보다 뛰어나다.

3. 스마트-시티 패러다임과 미들웨어

본 사업단에서는, 선행 연구에서, 유토피아라는 3 티어 아키텍처를 개발하여 사용해오고 있다. 이 내용은 이미 논문으로 일부 발표된바 있다.[10]

그림 1은 유토피아에서의 지능형 데이터 처리에 중심을 두어 메커니즘을 설명한 구조도이다. 유토피아의 3티어는 서로 연동하여 작동하며, 스마트-시티를 효율적이



(그림 1) 유토피아 시스템
(Figure 1) UTOPIA Smart-city system

고, 효과적으로 관리할 수 있게 한다. 3티어는 스마트-시티 포탈 티어, 스마트-시티 미들웨어 티어, 스마트-시티 인프라스트럭처 티어로 구성된다. 스마트-시티 포탈 티어는 소울의 여러 기능들을 이용하여 스마트-시티 서비스를 사용자에게 제공한다. 이 중, 소울 미들웨어는 스마트-시티 인프라스트럭처를 통해 들어오는 다양한 데이터를 지능적으로 처리하는 기능을 제공한다.

다음 절부터 설명되는 본 논문의 핵심내용을 이해하기 위하여, 본 절에서는, 그림 1을 이용하여 유토피아에서의 지능형 데이터 처리에 중심을 두어 작동 메커니즘을 아래와 같이 간략하게 설명하고자 한다. 공통-장치-인터페이스-레이어는 인프라 영역의 센서로부터 데이터를 받아 적절한 처리를 하면서 동시에 이를 스마트-컴퓨팅-레이어로 전달하는 역할을 수행한다. 다양한 종류의 센서와 통신 프로토콜을 지원한다. 다양한 종류의 유비쿼터스-센서-네트워크를 지원하는 공통 인터페이스를 제공한다. 스마트-컴퓨팅-레이어는 스트림-리즈닝을 위한 기능들을 수행하는 중요 계층의 하나이다. 상황변환기, 상황추론기, 상황저장소, 상황관리기가 주요 구성요소이다. 상황변환기는 스트림-리즈닝을 수행하기 위하여 전송 데이터를 컨텍스트 온톨로지 모델을 따르는 RDF/OWL 포맷의 데이터로 변환하고 저장하는 일을 한다. 필터링 규칙에 따라 필터링을 수행한다. 상황추론기는 스트림-리즈닝을 수행하면서 미리 정의된 규칙에 따라 추론한다. 이 때, 스트림-리즈닝의 실시간 처리를 위하여 유비쿼터스-코어-컴퓨팅-레이어의 클라우드-컴퓨팅 기술을 사용한다. 이 내용은 다음 절에서 상세히 설명되어진다. 소울은 클라우드-컴퓨팅을 구현하기 위하여 초기의 자체 개발 방식에서, 오픈-소스-소프트웨어인 OpenNebula [11]와 Haizea [12]를 기반 소프트웨어로 사용하는 방식으로 변환되었다. 이는 앞으로의 업그레이드, 널리 호환성 있게 사용하게 하려는 목적 등등을 고려한 결과이다. 때맞추어 효용성 있는 관련 오픈-소스-소프트웨어들이 개방되어 이것이 가능해졌다. 상황저장소는 컨텍스트 온톨로지 모델, 컨텍스트 인스턴스, 규칙, 그리고 추론된 컨텍스트를 저장한다. 상황관리기는 유비쿼터스-코어-컴퓨팅-레이어와의 인터페이스 역할을 수행하며, 유비쿼터스-코어-컴퓨팅-레이어의 서비스-브로커로부터의 질의를 해석하고, 질의를 수행하는 등의 전반적인 관리 업무를 수행한다. 컨텍스트 온톨로지 모델과 규칙 파일은 시스템이 초기화 될 때 메모리에 미리 탑재 된다. 컨텍스트 인스턴스는 실행시간에 동적으로 메모리에 탑재된다. 소울은 컨텍스트에 대한 연속질의를 지원한다.

4. 실시간 스트림-리즈닝 처리

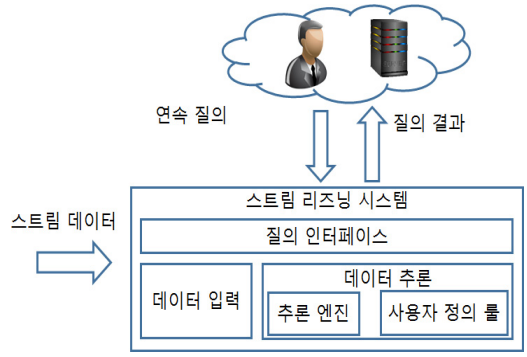
4.1 스트림-리즈닝

본 사업단은 스트림-리즈닝에 사용가능한 방법론과 시스템을 연구 개발하였다. 아파치 카프카 (Apache Kafka) [13]와 아파치 스톰(Apache Storm)을 본 시스템에서 사용할 최적의 오픈-소스-소프트웨어로 선정하여서, 본 연구단에서 기존에 개발했던 관련 모듈들을 이것으로 교체하였다. 이는 추후의 업그레이드와 보편성을 높이기 위함이다. 개발된 시스템은 스트림 데이터의 양이 시시각각으로 증감할 때, 이를 유연하게 소화할 수 있는 확장성과 실시간으로 데이터를 처리하기 위하여 충분히 빠른 응답시간이라는 요구조건을 만족해야 한다. 개발된 시스템은 이들을 만족시킨다.

본 사업단에서 개발한, 스마트-시티에서 사용하기 위한, 스트림-리즈닝 시스템은 그림 2와 같이 데이터 입력, 데이터 추론, 질의 인터페이스 등을 주요 구성요소로 하고 있다. 데이터 입력 구성요소는 스트림 데이터를 전달받아서, 데이터 추론 구성요소에 실시간으로 전달하는 역할을 한다. 질의 인터페이스 구성요소는 외부 시스템에서 스트림-리즈닝 시스템에 추론된 정보를 요구할 수 있는 인터페이스를 제공한다. 외부 시스템에서 연속적으로 질의를 할 경우, 스트림-리즈닝 시스템은 외부 시스템에서 연속적으로 청구한 질의를 연속적으로 처리하여 추론된 정보를 반환한다.

데이터 입력 구성요소에서 대용량의 데이터를 연속적으로 실시간에 전송받아 처리하기 위하여 아파치 카프카를 사용한다. 카프카는 대용량 데이터 처리에 특화된 소프트웨어라는 점과 스트림으로 연속되어 전송되는 데이터를 처리할 수 있다는 점과 실시간 처리가 가능하다는 점, 기존 메시징 프로토콜인 AMQP 프로토콜이나 JMS API 등과 비교할 때 오버헤드가 적은 프로토콜을 사용하고 있어서 전송 지연시간이 적다는 점, 확장성이 있다는 점과 고가용성의 특성이 있다는 점, 클러스터로 동작되는 점 등의 장점을 고려하여 카프카를 채택하였다.

아파치 스톰은 분산 처리를 하면서, 실시간 분석을 지원한다. 입력되는 데이터 스트림을 스파우트(Spout), 볼트(Bolt), 토폴로지와 로직을 사용하여 처리한다. 스파우트는 토폴로지를 사용하여 외부로 부터 전송되는 데이터 스트림을 받는 역할을 한다. 볼트는 토폴로지를 사용하여, 작업을 처리한다. 처리할 작업을 여러 개로 분리하여 여러 개의 볼트를 배치할 수 있다. 이러한 구조와 처리 방식은 빅-데이터의 실시간 스트림-리즈닝 처리에 매우 적합하다.

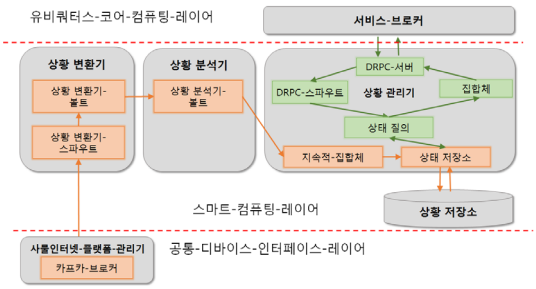


(그림 2) 스트림-리즈닝 시스템의 작동 메커니즘
(Figure 2) Working mechanism of our stream reasoning system

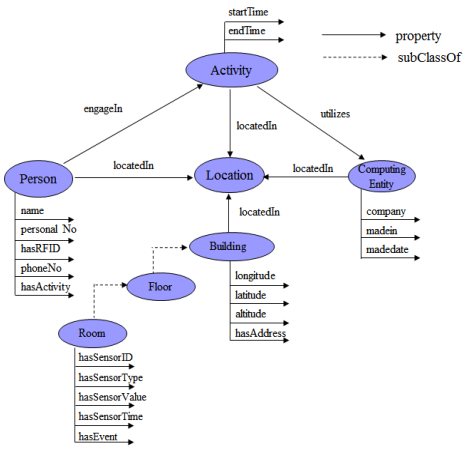
4.2 소울 실시간 스트림-리즈닝

그림 3은 본 사업단이 개발한 미들웨어인 소울에 구현된 실시간 스트림-리즈닝 시스템의 작동 메커니즘을 보여주고 있다. 소울의 공통-장치-인터페이스-레이어 상에 있는 사물인터넷-플랫폼-관리기에 카프카-브로커를 배치하여 스마트-컴퓨팅-레이어의 상황변환기에 데이터를 전송할 수 있도록 하였다. 상황변환기는 상황변환기-스파우트와 상황변환기-볼트를 사용하여 실시간 스트림-리즈닝 관련 데이터를 처리한다. 상황변환기-스파우트는 카프카-컨수머(Kafka Consumer)를 스톰(Storm)을 사용하여 구현한 것으로서, 사물인터넷-플랫폼-관리기로부터 전송되는 데이터스트림을 받아들이는 역할을 한다.

상황변환기-볼트는 전송받은 데이터스트림을, 추론이 가능하도록, OWL 온톨로지 모델을 이용하여, OWL 온톨로지 인스턴스로 변환하는 기능을 제공한다. 본 사업단에서 화재사고처리를 위한 스마트 서비스에 사용할 목적으로 제작한, OWL 도메인 온톨로지의 일부를 그림 4로 표시하였다. 상황추론기는 OWL 온토롤지 인스턴스와 사용자 정의 규칙을 이용하여 추론을 수행하며, 상황분석기-볼트를 포함한다. 상황분석기-볼트는 아파치-제나-프레임워크(Apache Jena Framework [14])를 이용하여 OWL 추론을 수행하며, OWL 추론기는 팰릿(Pellet [15])을 사용한다. 여기에서 사용되어지는 사용자 정의 규칙은 사용자의 입력을 받아 실행시간 중에 동적으로 변경이 가능하도록 제작되어 있다.



(그림 3) 소울의 스트림-리즈닝 시스템
(Figure 3) The stream reasoning system in SOUL



(그림 4) 화재사고처리를 위한 도메인 온톨리지의 일부
(Figure 4) A part of domain ontology for a fire accident management in smart cities

상황관리기는 상황추론기에서 수행된 추론 결과를 상황저장소에 전송하며, 상위 레이어인 유비쿼터스-코어-컴퓨팅-레이어의 서비스-브로커와의 인터페이스 기능을 수행한다. 각 노드들의 지속적-집합체(Persistent aggregate)는 처리된 추론 결과를 상태에 전달하는 역할을 한다. 상태 저장소는 DRPC(Distributed Remote Procedure Call)에서 전달되는 연속질의를 처리하기 위하여 상태를 저장하는 메모리에 위치하는 저장소이다. 스트림-리즈닝 할 때에 연속질의를 연속되는 추론의 결과에 적용하기 위해서는 각각의 순간에 대한 상태가 저장되어 있어야 한다.

스톰(Storm)에서 연속질의는 DRPC기법을 이용하여 처리한다[16]. DRPC-서버는 스톰(Storm)에서 사용하는 토폴로지가 RPC 를 수행할 수 있게 해주는 서버로써, 서비스-브로커에게서 명령을 받고, 처리된 결과를 서비스-브로커에 반환하는 역할을 수행한다. DRPC-스파우트는 DRPC-서버로 부터 받은 질의를 상태-질의에 전달한다. 전달된 질의를 이용하여 상태-질의는 현재 상태 값에 대한 질의를 하고, 그 결과를 받아 집합체에 전달한다. 집합체는 전달받은 결과값을 DRPC-서버에 전달하고, 최종적으로, DRPC-서버는 그 연속질의의 결과값을 서비스-브로커에 전달한다. 상황저장소는 상태 값들을 저장하는 저장소이며, 추론된 온톨로지 인스턴스를 저장하는 곳이다. 서비스-브로커는 유비쿼터스-코어-컴퓨팅-레이어에 속해 있고, 스마트-컴퓨팅-레이어와의 인터페이스 역할을 수행한다. 스마트-시티 어플리케이션은 필요한 상황정보를 획득하기 위하여 서비스-브로커를 통해서 연속질의를 요청하고, 그 결과를 받게 된다.

5. 성능평가

지금까지 본 사업단이 개발한 실시간으로 스트림-리즈닝 처리를 하는 방법과 이를 소울 미들웨어에 구축한 시스템에 대해서 서술하였다. 이번 장에서는 이것들을 바탕으로 하여 실험환경을 구축하고, 실제 환경을 시뮬레이션한 부하를 사용하여서 성능평가 실험을 한 결과와 성능분석 내용을 소개한다.

5.1 성능평가실험 환경

클러스터 방식의 클라우드-컴퓨팅을 이용하여 성능평가 실험을 하였다. 실험에 사용된 클러스터는 총 16대이다. 표 1과 표 2는 실험환경을 정리하여 보여준다.

CPU는 Intel core i5 760 2.8Ghz, 램은 DDR3 8GB, 하드디스크는 500GB 7200rpm, 네트워크는 1Gbps 이더넷(Ethernet)을 이용하였고, 운영체제는 CentOS 6.7 64bit 서버를 사용하였다. 클러스터 노드 3대에 카프카 브로커(Kafka Broker)와 아파치 주키퍼(Apache Zookeeper)를 같이 설치하고, 님부스(Nimbus)를 별도로 설치하였으며, 12대에 스톰 수퍼바이저(Storm Supervisor)를 설치하였다.

(표 1) 각 노드의 사양.
(Table 1) Specification of each Node.

종류	사양/버전
중앙처리장치	Intel core i5 760 2.8Ghz (Quad Core)
램	DDR3 8GB
하드디스크	500GB 7200rpm
네트워크	1Gbps Ethernet
운영체제	CentOS 6.7 64bit server edition

(표 2) 시스템 구성
(Table 2) system Configuration

종류	대수
총 클러스터의 노드 수	16
카프카 브로커/주키퍼 설치된 노드	3
DRPC 서버/님부스 설치된 노드	1
스톰 수퍼바이저 설치된 노드	12

5.2 사용된 부하 (Workload)

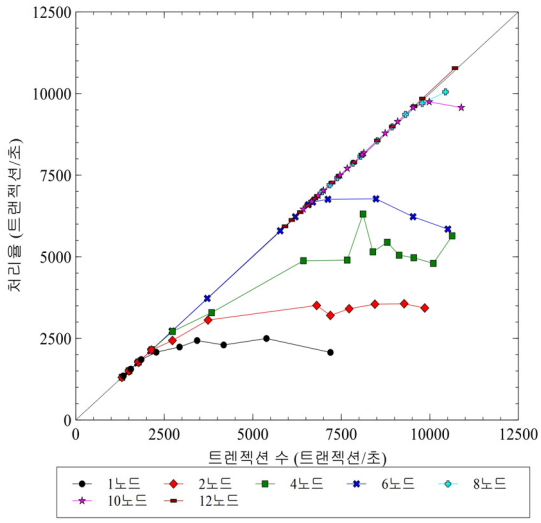
본 실험에서는 인공부하 (Artificial Workload)를 사용하였다. 본 사업단은, 센서로 부터의 전송되어지는 사물인터넷 스트림 데이터를 실제와 유사하게 발생시키는 스트림 데이터 생성기를 개발하였다. 이를 사용하여, 스마트-시티에서 발생하는 실제 부하 (Real Workload)를 시뮬레이션 하는 인공부하를 발생시켜 실험에 사용함으로서, 성능평가실험의 사실도와 충실도를 높였다. 이 실험에서 사용된 데이터는 센서 사물인터넷 데이터로서 트랜잭션당 200bytes의 고정된 크기 값을 가지고 있다. 매 실험 당 백만건까지 트랜잭션을 발생시켜서 소울 미들웨어에 전송하였다.

5.3 실험수행

그림 2에 도시된 시스템을 이용하여, 스트림 데이터를 입력받아서 화재사고가 발생하였을 때 위험 지역을 추론하는 실험을 수행하였다. 이 경우에, 실시간 스트림-리즈닝이 제대로 수행되는지를 확인하고 최적의 실험수행환경도 알아내고자 하였다. 실제 성능을 측정함에 있어서, 응답시간, 즉, 처리시간과 매초당의 데이터 처리량을 측정하였다. 시스템의 성능을 증가시키기 위하여, 시스템을 구성하는 노드의 개수를 증가시키는 방법론을 사용하였다. 노드의 개수는 1개로부터 시작하여, 2개 노드, 그 후는 2개 단위로 증가시켰다. 최대 12대까지를 처리노드로 사용하였다. 각 노드마다 독자적인 기억장치를 가지는 분산시스템 구조로 시스템을 구축하였다. 이를 클라우드 플랫폼을 통하여 실험에 사용하였기에, 본 실험에서는, 메모리의 공유에서 오는 문제점은 없다.

5.4 실험결과 및 성능분석

그림 5는 센서로 부터의 사물인터넷 스트림 데이터 전송량과 시스템에서 사용한 노드 개수를 변화시켰을 때에



(그림 5) 성능평가 실험 결과.
(Figure 5) Performance evaluation result.

실제 측정한 전체 처리시간의 변화를 나타낸 그래프이다. 초당 유입되는 메시지의 개수가 초당 처리되는 작업의 개수와, 즉, 처리율(Throughput)과 같을 때, 실시간 처리가 제대로 되고 있는 상황이다. 따라서, 표시된 정비례 직선은 실시간 처리가 제대로 될 경우의 이상적인 그래프이다. 너무 많은 메시지가 유입되어 실시간 처리가 되지 않는 경우에는 점차로 표시된 정비례 직선에서 멀어지는 포물선을 그리게 되는 것이 바람직한 경우이다.

1개의 노드를 사용했을 때, 초당 2500개의 메시지가 센서로부터 유입될 때까지, 실시간 처리가 잘 진행되었다. 즉 정비례 직선형태를 유지하였다. 그 이상의 메시지가 유입되면, 점차로 포화현상이 나타나서, 실시간 처리율이 떨어지기 시작하였다. 대체로 포물선 형식을 나타내면서 처리율이 떨어지기 시작하였는데, 이는 시스템이 과부하에서도 바람직하게 작동한다는 것을 보여준다. 일종의 스트레스 테스트 (Stress Test)에도 시스템이 잘 적응함을 보여주고 있다.

노드 개수를 2개에서 4개, 6개, 8개, 10개, 12개로 점차로 증가하여 감에 따라, 성능은 비례에 가깝게 좋아졌다. 4개의 노드의 경우, 스트레스 테스트를 할 때, 초당 유입되는 메시지가 8500건 근처에서 그리고 11000건 근처에서, 예상 밖으로, 포물선보다 좋은 성능을 보였다. 노드가 10건일 때에 11500건 이상의 메시지 시스템에 유입되면, 예상 밖으로, 8노드의 경우보다 낮은 초당 처리률을 보였

다. 10노드의 경우에 스트레스 테스트에서 8노드 보다 더 불안정함을 보인 것이다. 이 모든 것을 종합하여 볼 때, 부하의 증가에 따른 시스템의 안정적인 운영을 표시하는 확장성 항목에 있어서, 우수한 성능을 보였다고 평가된다. 성능평가실험에서 진행된 결과를 보면, 용량계획(Capacity Plannning)이 적절히 이루어졌음을 알 수 있다. 따라서, 이 실험의 결과를 이용하면, 각 상황에 맞는 적절한 시스템을 구축하여 운영할 수 있다.

전반적으로 구축된 실험시스템은 잘 작동함을 확인할 수 있었으며, 본 사업단에서 개발하여 구축하고, 본 논문에서 소개하는 실험시스템은 성공적으로 작동하였으며, 여러 성능평가 사항에서, 매우 우수한 성능을 보임을, 그림 5를 통하여 알 수 있다. 이런 면에서, 성능평가실험은 매우 성공적이었다.

6. 결 론

본 사업단에서는, 스마트-시티의 각종 센서들로부터 전송되어지는 사물인터넷 빅-데이터를 사용하여 실시간으로 스트림-리즈닝하는 데 필요한 기법과 시스템을 개발하였고, 본 논문에서 그 내용을 소개하였다. 우리의 시스템은 실시간 클라우드-컴퓨팅 기술을 기반으로 하는 실시간 분산처리 방법론을 사용하고 있음을 본 논문에서 먼저 상세히 소개하였다. 이 방법론을 선행연구에서 개발한 스트림-리즈닝 방법론에 융합시킨 후, 이를 스마트-시티 미들웨어 소울에 구현하였음을, 본 논문에서 설명하였다. 마지막으로, 구현된 시스템의 성능을 평가하여, 이 시스템이 정상적으로 잘 작동하고 있음과 여러 가지 우수한 성능을 보임을 증명하였다.

성능평가를 통하여, 주어진 환경에서 최적의 시스템을 선택할 수 있도록, 방법론을 제시하였다. 이 방법론은 용량계획에도 유용하게 쓰일 수 있음을 설명하였다. 이것은, 실제 시스템 구축 시, 클러스터나 클라우드 시스템의 컴퓨팅 파워의 용량 등등을 고려하는데 매우 유용한 정보와 자료로 사용될 수 있다.

성능이상의 부하를 걸어 줌으로서, 스트레스 테스트를 수행하였으며, 이를 통하여, 과부하시, 본 연구에서 개발된 실시간 스트림-리즈닝 시스템이 어떻게 작동하는지도 알 수 있도록 방법론을 제시하고 실제 결과를 보여주었다. 또한, 센서의 개수가 늘어남에 따라, 전송데이터가 증가하기 때문에, 이를 실시간으로 성공적으로 처리하기 위하여, 컴퓨팅 파워가 부하상황에 따라, 매끄럽고 자연스럽게 확장될 수 있는 특성이 요구된다. 이것이 잘 만족될 수 있음을 성능평가에서 보여주었다.

향후 연구에서는, 스마트-시티 어플리케이션을 사용할 때에, 사용자가 질의 결과를 어느 정도의 응답시간 내에 획득할 수 있는지에 대해서 알아보고자 한다. 또한 연속질의들 만에 대한 지연시간을 측정하는 실험도 수행하고자 한다. 이 사항들은 본 연구단의 시스템에 필요한 평가요소들이기 때문이다.

참 고 문 헌(References)

[1] Y. W. Lee, “Smart-city”, European Union Parliament Seminar, May 2013, [Online], Retrieved June 2016 from http://www.europarl.europa.eu/document/activities/cont/201305/20130514ATT66084/20130514ATT66084EN.pdf.

[2] E. D. Valle, S. Ceri, F. v. Harmelen, D. Fensel, “It’s a Streaming World! Reasoning upon Rapidly Changing Information”, IEEE Intelligent Systems, Vol. 24, pp. 83 - 89, 2009. http://dx.doi.org/10.1109/MIS.2009.125

[3] Apache Storm, [Online], Retrieved Jun. 2016 from http://storm.apache.org/.

[4] Apache S4, [Online], Retrieved June 2016 from http://incubator.apache.org/s4/.

[5] Apache Samza, [Online], Retrieved Jun. 2016 from http://samza.incubator.apache.org/.

[6] Apache Spark, [Online], Retrieved Jun. 2016 from http://spark.apache.org/.

[7] J. Hoeksema, S. Kotoulas, “High-performance Distributed Stream Reasoning using S4”, Proc. The First International Workshop on Ordering and Reasoning (OrdRing 2011), pp. 1-12, 2011. https://goo.gl/z9Mu35

[8] X. Chen, H. Chen, N. Zhang, J. Huang, W. Zhang, “Large-Scale Real-Time Semantic Processing Framework for Internet of Things”, International Journal of Distributed Sensor Networks, Vol. 11, No. 10, pp. 1-11, 2015. http://dx.doi.org/10.1155/2015/365372

[9] A. I. Maarala, X. Su, J. Riekki, “Semantic Reasoning for Context-aware Internet of Things Applications”, IEEE Internet of Things Journal, Vol. 3, No. 4, pp. 1-13, 2016. http://dx.doi.org/10.1109/JIOT.2016.2587060

[10] H. S. Jung, C. S. Jeong, Y. W. Lee, P. D. Hong, “An Intelligent Ubiquitous Middleware for U-City: SmartUM”, Journal of Information Science and Engineering, Vol. 25, pp. 375-388, 2009. http://dx.doi.org/10.1.1.423.772

[11] OpenNebula Homepage, [online], Retrieved Jun. 2016 from http://opennebula.org/.

[12] B. Sotomayor, R. S. Montero, I. M. Llorente, I. Foster, “Virtual Infrastructure Management in Private and Hybrid Clouds”, IEEE Internet Computing, Vol. 13, Issue 5, pp. 14-22, 2009. http://dx.doi.org/10.1109/MIC.2009.119

[13] J. Kreps, N. Narkhede, J. Rao, “Kafka: a Distributed Messaging System for Log Processing”, Proc. NetDB workshop (NetDB 2011), 2011. http://dx.doi.org/10.1.1.233.1726

[14] Apache Jena, [online], Retrieved Jun. 2016 from https://jena.apache.org/.

[15] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, Yarden Katz, “Pellet: A practical OWL-DL reasoner”, Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 5, Issue 2, pp. 51-53, 2007. http://dx.doi.org/10.1016/j.websem.2007.03.004

[16] Storm Distributed RPC, [online], Retrieved Jun. 2016 from http://storm.apache.org/releases/current/Distributed-RPC.html.

◐ 저 자 소 개 ◑

윤 창 호(Chang Ho Yun)

2008년 서울시립대학교 전자전기컴퓨터공학부 (공학사)
2010년 서울시립대학교 전자전기컴퓨터공학과 (공학석사)
2010년~현재 서울시립대학교 전자전기컴퓨터공학과 박사과정
관심분야 : 클라우드 컴퓨팅, 스마트시티, 시맨틱웹, ICT융합시스템, IoT, 인터넷
E-mail : touch011@uos.ac.kr

박 종 원(Jong Won Park)

2009년 서울시립대학교 전자전기컴퓨터공학부 (공학사)
2011년 서울시립대학교 전자전기컴퓨터공학과 (공학석사)
2011년~현재 서울시립대학교 전자전기컴퓨터공학과 박사과정
관심분야 : 인터넷, 클라우드 컴퓨팅, 시스템 소프트웨어, 스마트시티, ICT융합시스템, IoT
E-mail : comics77@uos.ac.kr

◐ 저 자 소 개 ◑

정 혜 선(Hae Sun Jung)

1992년 명지대학교 전자계산학과 (공학사)
2001년 고려대학교 전자컴퓨터공학과 (공학석사)
2011년 고려대학교 전자컴퓨터공학과 (공학박사)
2015년~현재 서울시립대학교 연구교수
관심분야 : 인터넷, 클라우드 컴퓨팅, 시스템 소프트웨어, 스마트시티, ICT융합시스템, IoT
E-mail : banyasun@uos.ac.kr

이 용 우(Yong Woo Lee)

1981년 서울대학교 전기&컴퓨터 (공학사)
1990년 영국 Univ of Edinburg (공학석사)
1997년 영국 Univ of Edinburg (이학박사)
1999년~현재 서울시립대학교 전자전기컴퓨터공학과 교수
E-mail : ywlee@uos.ac.kr