김일성종합대학학보
정보과학
주체110(2021)년
제67권 제3호

기계번역봉사체계에서 Node.js를 리용한 번역봉사기응답성능의 개선방법

김룡혁, 현경일

경애하는 **김정은**동지께서는 다음과 같이 말씀하시였다.

《첨단돌파전은 현대과학기술의 명맥을 확고히 틀어쥐고 과학기술의 모든 분야에서 세계를 앞서나가기 위한 사상전, 두뇌전입니다.》

Node.js는 최신Web기술의 주류를 이루고있다. 현재 대규모자료를 처리하는 봉사를 Node.js환경에 신속히 적용하고 종전보다 성능이 향상된 봉사환경을 제공하기 위한 연구[1, 2]가 활발히 진행되고있다.

선행연구들에서는 사용자들의 다량의 동시번역요청에 대한 실시간응답특성을 개선하기 위한 보다 합리적인 Web봉사플래트홈(platform)을 선택하는 방법을 제안하였지만 응답성능을 제고하는데서 일련의 결함을 가지고있다.

론문에서는 Node.js를 리용하여 동시번역요청의 실시간응답특성을 개선하기 위한 분산병렬처리방법을 제안하였다.

1. Node.js를 리용한 기계번역봉사체계설계

이전의 봉사기들은 동시에 다량적으로 발생하는 요청들을 처리하기 위하여 스레드(thread)를 리용한 처리방식을 적용하였다. 봉사기의 입출력명령처리에서 스레드처리는 Web응용프로그람을 관리하는 가장 좋은 방법으로 되여왔다. 그러나 동기방식의 스레드를 동시에 생성하여 명령을 처리하는 다중스레드를 개발리용하는데 품이 많이 드는 제한성을 가지고있다.

Node.js의 중요한 특징은 봉사기측에서 실행되는 망응용프로그람으로 사건구동방식의 비동기식입출력처리를 진행한다는것이다.[1]

Socket.io는 의뢰기와 봉사기사이의 쌍방향통신기능을 제공하는 Node.js확장모듈로서 Web응용프로그람의 처리능력과 확장가능성을 제고해주며 직결식유희(Online Game)나 실시간대화와 같은 실시간Web응용프로그람개발에 적합하다.

기계번역봉사체계의 번역봉사에서 실시간성을 보장하기 위해 Node.js환경에서 socket.io 모듈을 리용한다.

기계번역봉사체계의 구성을 그림 1에 보여주었다.

기계번역봉사체계의 매 요소들을 구체적으로 보면 다음과 같다.

① 소케트봉사기

Web봉사기와 번역봉사기사이의 문자렬통신을 보장한다. 소케트봉사기는 어종별로 창조되며 관리자페지에서 관리된다.

② Socket.io봉사기

Websocket규약을 리용하여 사용자들의 번역요청을 실시간적으로 처리해주는 기능을 수행한다.

Socket.io봉사기는 쎄션(session)관리자와 사용자쿠키(cookie)인증처리를 진행한다.

③ Web봉사기

Web봉사기는 Web열람기의 요청에 대한 http봉사를 진행한다.

④ 자료기지봉사기

사용자정보와 번역리력 등 체계관리에 필요한 자료들을 관리한다.

⑤ 쎄션관리자

사용자대화관리를 진행한다.

⑥ 접속관리자

접속관리자는 다량의 동시번역요청에 대하여 Web봉사기와 번역봉사기와의 병렬통신을 보장해주는 처리를 진행한다.

⑦ Web열람기

사용자와의 호상작용을 처리하는 대면부이다.

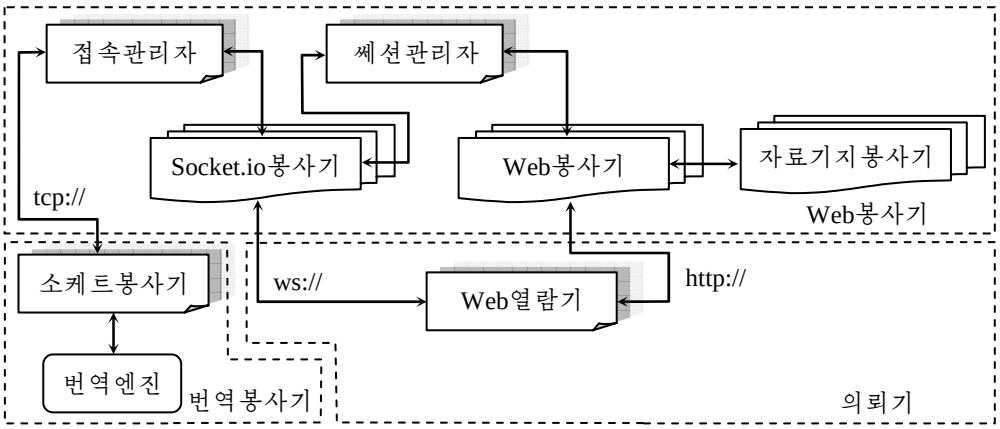

그림 1. 기계번역봉사체계의 구성

2. 병렬처리를 리용한 번역봉사응답성능개선

기계번역봉사체계에서 어떤 번역요청 S에 대한 처리시간은 요청처리시간 $Q_{\mathrm{Tran}}(S)$와 Web봉사기내에서의 지연시간 $Q_{\mathrm{Delay}}(S)$ 합으로 계산한다.

$$Q_{\mathrm{Time}}(S) = Q_{\mathrm{Tran}}(S) + Q_{\mathrm{Delay}}(S)$$

여기서

$$Q_{\mathrm{Tran}}(S) = T_{\mathrm{tran}} + T_{\mathrm{sock}} + T_{\mathrm{ws}}$$

이다. 그리고 T_{tran}은 번역봉사기에서 번역에 소비되는 시간이고 T_{sock}는 Web봉사기와 번역봉사기와의 소케트통신시간이며 T_{ws}는 의뢰기와 봉사기사이의 Websocket통신시간이다.

의뢰기와 봉사기사이의 자료전송에 리용되는 대역폭이 충분한 경우 동시번역요청처리시간은 번역처리시간 T_{tran}과 지연시간 $Q_{\mathrm{Delay}}(S)$에 의해 결정된다.

한편 기계번역봉사체계내에서 동시번역요청들의 모임 $S = \{S_1, S_2, \cdots, S_n\}$에 대한 처리흐름은 순차적으로 혹은 병렬적으로 구성할수 있다.

동시번역요청들에 대한 순차적처리흐름을 그림 2에 보여주었다.

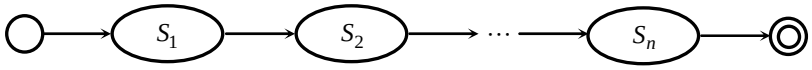

그림 2. 동시번역요청들에 대한 순차적처리흐름

동시번역요청들에 대한 병렬처리흐름을 그림 3에 보여주었다.

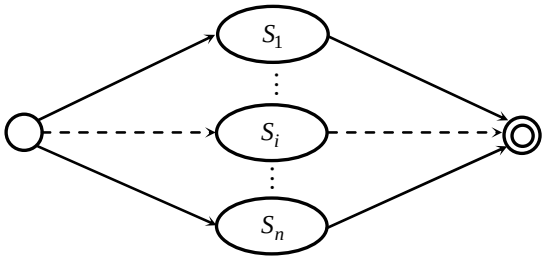

그림 3. 동시번역요청들에 대한 병렬처리흐름

순차적인 번역처리흐름에서는 n개의 번역요청(S_i)들이 순차적으로 처리된다. 이때 처리 S_i 는 처리 S_{i+1} 이 시작되기 전에 완료되여야 하므로 전체적인 봉사응답시간은

$$Q_{\text{Time}}(S) = \sum_{i=1}^{n} Q_{\text{Time}}(S_i)$$

$$Q_{\text{Delay}}(S) = \sum_{i=1}^{n} Q_{\text{Delay}}(S_i)$$

와 같이 결정된다.

병렬적인 처리흐름은 n개의 병렬처리들로 구성되며 매 번역요청 S_i 는 호상 독립적으로 동시에 처리된다.

전체적인 봉사응답시간은 다음과 같이 결정된다.

$$Q_{\text{Time}}(S) = \text{MAX}_{1\le i\le n}(Q_{\text{Time}}(S_i))$$

$$Q_{\text{Delay}}(S) = \text{MAX}_{1\le i\le n}(Q_{\text{Delay}}(S_i))$$

병렬처리흐름개수 n은 봉사기가 리용할수 있는 RAM의 크기에 비례한다.

Node.js에서는 병렬처리를 위해 CPU핵심처리기분할방식(cluster)과 분기(fork)방식을 지원한다. cluster방식에서는 프로쎄스들사이의 통신문제가 제기되므로 child_process에 의한 분기방식을 리용한다. 이때 매개 자식프로쎄스들이 소비되는 기억기크기는 5～6M정도이다.

3. 실험 및 성능평가

기계번역봉사체계를 Node.js의 다중프로쎄스기술을 리용하여 n개 요청에 대하여 n개의 독립적인 자식프로쎄스들을 창조하고 부모프로쎄스와 통신하도록 구성한다.

동시요청응답분포특성그라프를 그림 4에 보여주었다. 그림 4에서는 병렬처리방식에서 200명의 사용자들이 한문장씩 동시번역요청을 보내는 경우의 응답특성을 보여주었다. 동시요청응답분포특성을 보면 1～3s내로 응답한 비률은 사용자수 200명가운데서 66%이며 4s이상 걸린 응답은 4%이다.

론문에서 제안한 방법이 2 000여명의 사용자들이 동시번역요청시 16GB의 기억기를 소비함으로써 이전의 Web봉사기들에 비하여 속도나 기억기리용에서 5배이상의 성능을 높인다는것을 알수 있다.

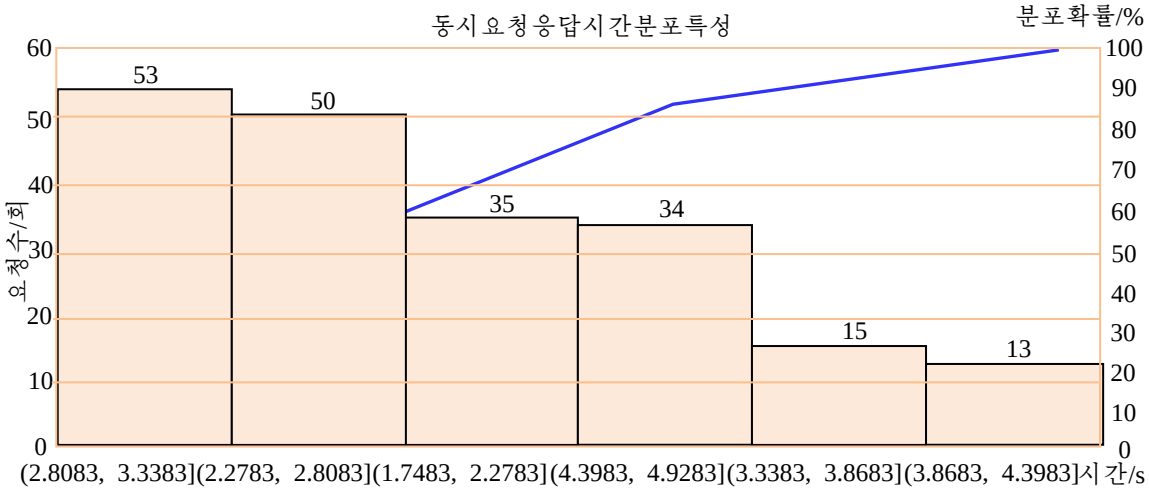

그림 4. 동시요청응답분포특성그라프

맺 는 말

Node.js의 비동기식통신모듈을 리용하여 대규모번역요청처리에 대한 요청응답성능을 개선하기 위한 한가지 방법을 제안하고 실험을 통하여 그 성능을 확증하였다.

참 고 문 헌

[1] 김일성종합대학학보(자연과학), 63, 9, 159, 주체106(2017).
[2] 黄志峰 等; 中国医疗设备, 33, 10, 23, 2018.

주체110(2021)년 5월 5일 원고접수

Improving Translation Service Response Performance with Node.js in Machine Translation Service System

Kim Ryong Hyok, Hyon Kyong Il

We proposed a parallel processing scheme to improve the request response performance for large-scale translation request processing using Node.js asynchronous communication module, and verified it in experiments.

Keywords: machine translation service system, node.js server, multi request