

카풀 및 그룹핑 기법을 이용한

유전자 서열 정렬 프로그램(FastA) 설계*

이성준^o 김재훈 정진원 이원태

아주대학교 정보통신전문대학원, 연세대학교 생화학과

{piero00^o, jaikim}@ajou.ac.kr {solwind,wlee}@spin.yonsei.ac.kr

Design of Gene Alignment Program(FastA) Using Carpool and Grouping Schemes

Sungjoon Lee^o Jai-Hoon Kim Jin-Won Jung Won-Tae Lee

Graduate School of Information and Communication, Ajou University
Department of Biochemistry, Yousei University

요약

생물정보학에서 사용되는 많은 프로그램들은 데이터베이스로 부터 방대한 양의 데이터를 검색하고 처리한다. 이러한 환경에서 사용자의 요청마다 데이터베이스를 검색하는 경우 사용자들의 대기 시간이 길어지고 시스템 용량을 초과한다. 이러한 데이터베이스 액세스의 문제점을 해결하기 위하여 카풀 기법과 그룹핑 기법이 제안되었다. 본 논문에서는 카풀 기법과 그룹핑 기법을 이용하여 유전자 서열 비교 프로그램인 FastA를 구현하였고 사용자 응답시간을 측정하여 프로그램의 성능을 높일 수 있음을 확인하였다.

1. 서론

인간유전체 지도의 완성으로 인간의 유전자와 유전자의 발현으로 생기는 단백질들의 구조와 기능에 관한 연구가 활발히 수행되고 있다. 이로 인해 생기는 많은 정보들이 데이터베이스로 구축되고 있다. 웹을 통해 공개된 생물학 관련 데이터베이스는 SwissProt[4], GenBank[5], EMBL[6] 등 헤아릴 수 없을 정도로 많다[3]. 이런 환경에서 사용자의 요청마다 데이터베이스를 검색하고 그 결과를 처리한다면 사용자들의 대기 시간이 길어지고 시스템 용량을 초과한다. 이러한 데이터베이스 액세스 문제를 해결하기 위해 선행 연구에서 카풀 방식[2]과 그룹핑 방식[2]을 제안하였다.

2. 관련 연구

2.1 단백질 서열 비교 프로그램 FastA

FastA[1]는 임의의 서열과 유사성을 가진 서열을 서열 데이터베이스로부터 찾는 강력한 소프트웨어이다. FastA[1]는 단백질 서열간의 비교를 위해 제작되었지만 염기 서열간의 비교도 가능하다.

2.2 카풀 방식 [2]

카풀 방식은 사용자의 요청이 오면 큐에 등록시키는 것이 아니라 현재 처리되고 있는 요청과 함께 비교 분석한다. 그러므로 사용자들은 요청을 처리받기 위해 대기하지 않아도 되는 장점을 갖는다. 생물정보학에서 사용되는 데이터베이스에서 각각의 데이터는 서로 의존성이 없으므로 데이터베이스에 있는 데이터들의 처리되는 순서와 상관없이 모든 데이터베이스의 데이터를 처리할 수 있다는 가정 하에서 이 방식은 유효하다.

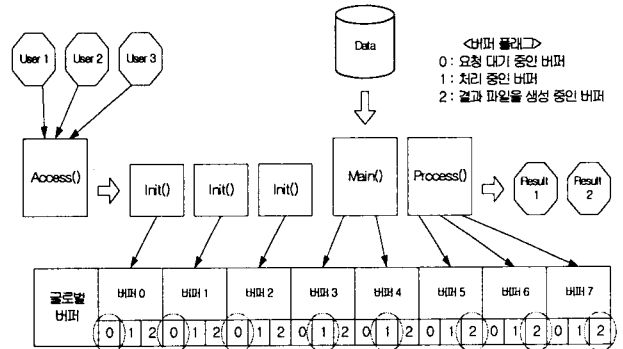
2.3 그룹핑 방식 [2]

이 방식은 사용자의 요청마다 데이터베이스를 처리하는 것이 아니라 일정 주기 동안 도착하는 사용자의 요청을 모아서 처리한다. 그러므로 데이터베이스를 접근하는 횟수를 줄일 수 있다.

3. FastA 프로그램의 수정

카풀 방식과 그룹핑 방식에서 동일하게 사용자에 대한 정보를 유지하기 위해 글로벌 버퍼를 만들었다. 그리고 사용자의 요청을 받는 부분, 사용자에 대한 정보를 초기화하는 부분, 서열을 비교하는 부분, 그리고 결과를 처리하는 부분을 멀티 스레드(Multi-thread)로 구현했다.

3.1 카풀 방식



[그림 1] 카풀 방식의 처리 과정

[그림 1]은 사용자 요청이 왔을 때 프로그램이 카풀 방식으로 처리하는 과정을 보여주고 있다.

글로벌 버퍼(global buffer)가 사용자의 요청에 대해 처리를 하는 핵심이다. 버퍼는 구조체로 되어있으며 사용자에 대한 정보를 갖고 있다. 버퍼의 상태는 플래그(flag)를 통하여 결정된다. 0은 요청 대기 중인 버퍼를 나타내며, 1은 처리중인 버퍼,

* 본 연구는 정보통신부 정보통신 선도기반 기술개발사업의 지원에 의하여 이루어진 것임.

2는 결과 파일을 생성중인 버퍼를 나타낸다. 그러므로 초기에 사용자는 플래그가 0인 버퍼를 이용할 수 있다. 각 쓰레드의 기능은 [표 1]에 나타내었다.

[표 1] 카풀 방식에서 각 쓰레드의 기능

쓰레드	기능
Access()	(1) 사용자의 요청이 발생하면 접속을 맺는다. (2) 그 요청에 대한 처리를 Init() 쓰레드에게 넘긴다. (3) 다른 사용자의 요청을 대기한다.
Init()	(1) 글로벌 버퍼에서 대기 중인 버퍼(0)를 찾아 사용자에게 할당한다. (2) 사용자에게 대한 정보를 초기화한다. (3) 해당 버퍼의 플래그(flag)를 1로 설정한다.
Main()	(1) 모든 글로벌 버퍼의 플래그(flag)를 검사한다. (2) 플래그(flag)가 1일 경우 서열 비교를 한다. (3) 글로벌 버퍼에 서열 비교가 안 끝난 사용자가 있으면 계속 데이터베이스를 읽는다. (4) 서열 비교가 끝난 버퍼의 플래그(flag)는 2로 설정한다.
Process()	(1) 모든 글로벌 버퍼의 플래그(flag)를 검사한다. (2) 플래그(flag)가 2일 경우 버퍼의 내용을 바탕으로 결과 파일을 생성한다. (3) 처리가 모두 끝나면 그 버퍼의 플래그(flag)를 0으로 설정한다.

[그림 2]는 사용자 요청이 왔을 때 프로그램이 그룹핑 방식으로 처리하는 과정을 보여주고 있다.

이 방식에서는 그룹을 정하여 정해진 주기 동안 그 그룹에 대해서 서열비교를 한다. 선행연구[2]의 시뮬레이션에서는 시간으로 주기를 정하였으나, 본 논문의 실험에서는 버퍼의 개수로 주기를 정하였다. 버퍼에 대한 처리 시간이 주기가 될 수 있기 때문이다. 그룹핑(2)는 버퍼 2개를 사용한 것을 나타내고 그룹핑(3)은 버퍼 3개를 사용한 것을 나타낸다.

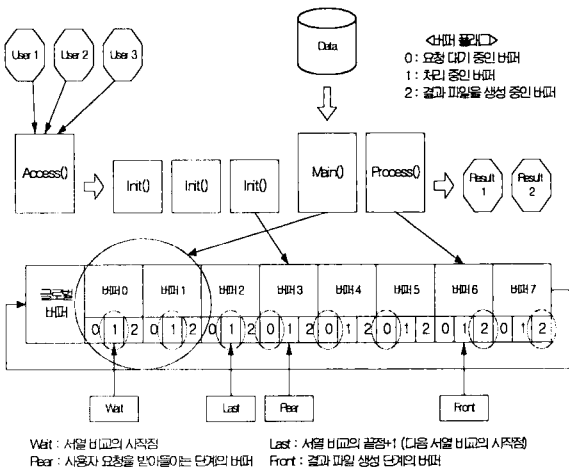
이 방식에서도 카풀 방식과 마찬가지로 글로벌 버퍼를 사용하고 플래그를 사용하여 버퍼의 상태를 나타낸다. 다른 점은 환경 큐와 4개의 포인터이다. 그룹을 만들기 위해선 포인터가 필요하고 글로벌버퍼의 데이터구조도 환경 큐여야 포인터 이동이 가능하다. 포인터는 4가지(Wait, Last, Rear, Front)가 있다. 먼저, Wait는 서열비교의 시작점을 나타낸다. 그리고 Last는 서열비교의 끝점의 다음 버퍼를 가리킨다. 그래서 그룹은 Wait에서 Last전까지의 버퍼로 정해진다. 사용자가 2명 이상일 때 그룹은 2개의 버퍼를 넘지 않고 적을 때는 그 수만큼만 그룹을 지정한다. 그리고 모자란 버퍼 수만큼 기다렸다가 다음 주기로 넘어간다. 다음 주기에서의 Wait는 현재의 Last가 가리키는 버퍼를 가리킨다. 여기서도 서열비교가 끝난 버퍼의 플래그는 2로 설정된다. Rear는 사용자의 요청을 대기하고 있는 버퍼를 가리킨다. 그리고 Front는 결과 파일을 생성해야 하는 단계에 있는 버퍼를 가리킨다.

각 쓰레드의 기능은 [표 2]에 나타내었다.

[표 2] 그룹핑 방식에서 각 쓰레드의 기능

쓰레드	기능
Access()	(1) 사용자의 요청을 받아들이고 그 처리를 Init() 쓰레드에게 넘겨준다. (2) 다른 사용자의 요청을 기다린다.
Init()	(1) Rear가 가리키는 버퍼를 사용자에게 할당한다. (2) 사용자에게 대한 초기화 작업을 한다. (3) 해당 버퍼의 플래그(flag)를 1로 설정한다.
Main()	(1) Wait와 Last가 만든 그룹을 모아서 서열 비교를 한다. (2) 글로벌 버퍼에 서열 비교가 (3) 서열비교가 끝난 버퍼들은 플래그(flag)가 2로 설정한다.
Process()	(1) Front가 가리키고 있는 버퍼의 내용을 바탕으로 결과 파일을 생성한다. (2) 결과 파일이 생성된 버퍼는 그 플래그가 0으로 설정된다.

3.2 그룹핑 방식



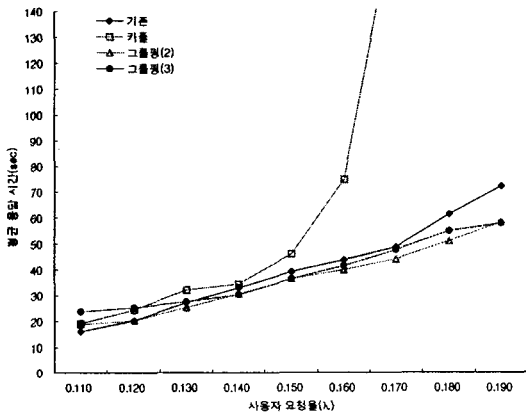
[그림 2] 그룹핑(2) 방식의 처리과정

4. 기존 방식과 카풀, 그룹핑 방식의 성능 비교

프로그램의 성능을 측정한 환경은 펜티엄4 1.6G Hz, 512Mbyte Ram 사양이다. 데이터베이스로는 단백질 서열 정보가 들어있는 pir.fasta[5]를 사용하였다. 사용자 요청 발생은 포아송 과정으로 가정하였고 150명의 사용자 요청을 처리하는 평균 응답 시간을 측정하였다.

[표 3] 각 방식의 사용자 요청에 따른 평균 응답 시간

사용자 요청율(λ)	기존 방식	카풀 방식	그룹핑(2) 방식	그룹핑(3) 방식
0.110	16.01087	19.41180	19.03407	23.80740
0.120	20.57100	24.73053	20.22167	25.33933
0.130	27.22827	32.39933	25.49093	27.70640
0.140	33.10587	34.66020	30.62227	30.42927
0.150	39.05287	46.06453	36.65993	36.56313
0.160	43.70120	74.99560	39.87807	41.39273
0.170	48.92513	180.22810	44.28227	47.54580
0.180	61.58640	-	51.08840	54.81040
0.190	72.28567	-	58.29073	58.06000



[그림 3] 각 방식에 따른 평균 응답 시간 그래프

[표 3]과 [그림 3]은 사용자 요청율의 변화에 대한 각 방식의 평균 응답 시간을 나타낸다. 사용자 요청율(λ)이 증가함에 따라 기존 방식과 카풀 방식의 평균 응답 시간이 증가하는 반면 그룹핑 방식은 상대적으로 평균 응답 시간의 변화가 완만하다. 사용자 요청율이 0.120 이전에는 기존 방식의 평균 응답 시간이 더 좋은 성능을 보여준다. 이 때는, 사용자가 빈번하게 도착하지 않으므로 오히려 기존 방식이 더 좋은 성능을 보였다. 카풀 방식의 경우 예상과는 달리, 기존 방식보다 좋은 성능을 보이지 못했다. 사용자 요청율이 0.140까지는 기존 방식과 비슷한 성능을 보이다가 0.150 이후부터는 급격하게 응답 시간이 증가한다. 카풀 방식의 경우 여러 사용자가 접속하면 같은 내용을 여러 명이 동시에 읽고 처리하므로 메모리를 많이 사용하게 된다. 그러한 메모리 오버헤드가 응답 시간이 좋지 않게 나오는 원인이라고 판단되어 이를 해결하는 중이다.

그룹핑 방식은 주기 시간이 있어서 도착한 사용자의 요청은 앞의 주기가 끝날 때까지 대기하고 있어야 한다. 그래서 일정한 대기 시간동안 기다려야한다. 사용자 요청율이 낮을 때는 기존 방식과 카풀 방식보다 높은 응답시간을 보이지만, 사용자 요청율이 높아지더라도 응답시간의 변화가 적다. 따라서 사용자 요청율이 0.130일 때는 그룹핑(2) 방식이 기존 방식과 카풀 방식보다 더 좋은 응답시간을 보여준다. 그룹핑(3) 방식은 주기가 길어서 사용자 요청율이 0.140 이전에는 다른 방식보다 안 좋은 성능을 보였지만 그 이후에는 기존 방식과 카풀 방식

보다는 좋은 성능을 보여주었다. 사용자 요청율이 0.140 이후 일 때, 카풀 방식의 평균 응답시간이 크게 증가했고 기존 방식은 평균 응답시간이 점점 증가했지만 그룹핑 방식은 변화가 적었다. 따라서 사용자 요청이 많을 때는 다른 방식보다 그룹핑 방식이 더 좋은 성능을 보여준다.

5. 결론 및 향후 계획

데이터베이스 액세스 방법에 있어서 선행연구에서 카풀 방식과 그룹핑 방식이 제안되었고, 시뮬레이션을 통하여 성능을 검증하였다[2]. 본 논문에서는 FastA 프로그램을 수정하여 두 방식을 구현하였고 기존 방식과 성능을 비교 측정하였다. 기존 방식은 사용자 요청율이 낮을 때에 다른 방식보다 더 좋은 성능을 보였다. 사용자 요청율이 낮을 때는 카풀이나 그룹핑 방식의 장점이 나타나지 않기 때문이다. 카풀 방식은 예상과는 달리, 기존 방식보다 좋은 성능을 보이지 못했다. 여러 사용자가 접속할 경우, 처리를 위해 메모리를 많이 사용하는 것이 오버헤드로 판단되어 이를 해결 중이다. 그룹핑 방식은 처음의 평균 응답 시간에서 큰 변동 없이 성능을 유지하였다. 주기 단위로 사용자 요청을 처리하므로 사용자 요청율이 낮을 때는 다른 방식보다 성능이 안 좋았지만 사용자 요청율이 높아 졌을 때는, 그룹을 이루어 한꺼번에 처리하는 방식으로 인하여 다른 방식들보다 더 좋은 성능을 보여주었다.

참고문헌

- [1] Pearson, W.R. and Lipman, D. , Improved Tools for Biological Sequence Comparison. Proc. Natl. Acad. Sci. USA, 85, pp. 2444-2448, 1988.
- [2] Min Jun Kim, Jai-Hoon Kim, Jin-Won Jung and Weontae Lee, "Data Sharing Model for Sequence Alignment to Reduce Database Retrieve," To appear in *International Conference on Computational Science 2003(ICCS'03)*, June 2003.
- [3] Hyun Seok Pa, "The Role of Bioinformatics in the Post Genomic Era," 대학내분비학회지, 제16권, 제1호, 2001.
- [4] SWISSPROT : <http://www.expasy.org/>
- [5] GENBANK : <http://www.ncbi.nlm.nih.gov/Genbank/>
- [6] EMBL : <http://www.embl-heidelberg.de/>