R 시스템에서 대량의 자료 처리

김기영, 김형석, 염헌영
서울대학교 컴퓨터공학부
{kykim, hskim, yeom}@dcslab.snu.ac.kr

Massive Data Process in R System

Kiyoung Kim, Hyeong Seog Kim, Heon Y. Yeom
School of Computer Science & Engineering
Seoul National University

요 약

R은 통계 계산과 그래픽을 위해 다양한 기능을 제공하는 언어 및 환경이다. R은 다양한 장점을 제공하지만, 대량의 자료 처리에 대한 고려가 되어 있지 않아서 사용자들이 불편을 겪고 있다. 이 논문에서는, 이러한 대량의 자료 처리 문제를 어떤 방법으로 해결할 수 있으며, 어떠한 추가적인 사항을 생각할 수 있는지 서술하고자 한다.

1. 서론

R[1]은 통계 계산과 그래픽을 위한 언어 및 환경이다. R은 다양한 통계적, 그래픽 기능을 제공하고, 사용하기 쉽게 구성되어 있다. R은 다음과 같은 기능들을 포함하고 있다.

- 효과적인 자료 관리 및 저장
- 배열과 행렬에 대한 계산 기능
- 자료 분석에 사용되는 다양한 기능들
- 자료 분석을 위한 그래픽 기능들
- 사용하기 쉽고 효과적인 개발 언어

R은 다양한 방면에서 사용되어지고 있다. 물리, 화학, 지구 환경 등 다양한 분야에서 R을 이용한 연구결과가 나오고 있다.[2] 대부분의 경우 실험 산출물을 분석하여 보다 쉽게 결과를 얻어내기 위한 도구로 R을 사용하고 있다.

하지만 R에도 문제점이 있다. 가장 큰 문제 중 하나는, R에서 대량의 자료를 읽어 들일 수 없다는 점이다. <표 1>은 64bit CPU에 메모리가 3GB인 컴퓨터에서 다양한 크기의 자료를 읽어 들일 때의 소요 시간과 소요 가상 메모리 사용량을 측정한 자료이다. 표에서 볼 수 있듯이, 외부에서 행렬 자료를 읽어 들일 때, 6배가 넘는 가상메모리 공간이 필요했고, 이 공간이 실제 메모리 크기를 넘어서는 순간 Swap File에 대량의 자료를 읽고 쓰게 됨으로써 성능이 매우 떨어지는 것을 확인할 수 있었다. 1.4G의 자료의 경우, 너무 큰 가상메모리를 요구하기 때문에, 32bit 컴퓨터에서는 일반적인 방법으로 자료를 읽어 들일 수 없었다.

기존 R을 사용한 결과 분석의 경우, 자료가 너무 많아서 R을 사용하기 힘들어지면 적절한 Sampling을 통해서 자료의 크기를 작게 한 후 분석하거나, 행렬을 잘게 쪼개서 계산하는 방법[3] 등을 사용해왔다. 하지만, Sampling을 사용할 경우 원래 결과와는 다른 결과가 나올 수도 있는 가능성이 있고, 행렬을 잘게 쪼개는 방법은 손이 많이 가고 실수하기 쉽다.

이 논문에서는 이러한 R의 문제점을 해결하기 위하여 어떤 방법을 사용하면 좋을지에 대한 아이디어를 제공하고 개선 여부에 대하여 이야기하고자 한다.

<표 1> R에서 외부자료를 읽어 들이는데 필요한 시간과 가상메모리 크기

Data Size	Loading Time	VM Size(loading)
200M	2 min	1.2G
400M	6 min	3G
1.4G	Over 1 week	12G

2. Temporary File을 이용한 자료 관리

이 논문에서는 임시 파일을 이용해서 외부 자료를 메모리에 읽지 않고 임시 파일에 저장해 두었다가 필요할 때 꺼내서 사용하는 방법을 제시하고자 한다.

R의 자료 형식은 0번부터 25번까지 크게 26가지의 형식으로 정의 되어 있다[4]. 형식은 NULL, symbol, vector 등 다양한 형식을 서술하고 있다. 기존의 형식을 그대로 사용하려면, 임시 파일을 이용해서 읽어 들인 자료와 그렇지 않은 자료 간에 구별을 하기가 힘들기 때문에 여기서는 새로운 자료 형식이 추가적으로 만들어서 사용하고자

한다. 현재 자료 형식을 위해 5bit를 쓰고 있으므로, 새로한 개의 자료형식을 늘려도 크게 문제가 되지 않는다. 이 새로운 자료 형식을 위해, R의 기존 자료 형식에 새로운 형식 'TFILESXP'(Temporary File Type)를 26번에 추가해 주도록 한다.

사용자가 특정 명령어를 통해 새로운 자료를 읽어 들이게 되면, 자료를 Stream 형식으로 읽어 새로운 임시 파일에 저장하고, 자료의 크기(행렬의 행 크기와 열 크기 등)와 임시 파일의 위치를 R에서 TFILESXP 형식의 해당 변수 공간 안에 저장한다. 임시파일에 저장할 경우에는 CSV(Comma Separated Value) 형식을 사용하거나 공백을 사용하여 데이터 간에 구별을 두는 방법을 사용한다. 자료를 요청할 경우, 원하는 자료의 행과 열 위치에 해당하는 자료를 파일로부터 읽어 들여서 값을 제공해 주는 방법을 사용하면 된다.

이러한 방법에는 세 가지 장점이 있다. 첫 번째, 자료의 크기 제한 문제가 해결된다. 기존의 메모리로 읽어 들이는 방식을 사용할 경우, 32bit 컴퓨터에서는 500M 이상의 파일을 읽어 들일 수 없었다. 하지만 파일을 사용해 대규모 자료를 관리할 경우, 그러한 자료 크기 제한 문제를 해결할 수 있다.

두 번째로, 자료를 읽어 들이는 시간을 줄일 수 있다. 만약 기존 자료 파일이 형식에 맞춰져 있을 경우, 새로이 임시 파일을 만들지 않고 기존 자료를 가지고 바로 사용할 수 있다. 이렇게 사용할 경우, 자료를 읽어 들이는 시간이 매우 줄어들게 된다.

마지막으로, R이 컴퓨터를 선점하는 일을 막을 수 있다. 기존의 R의 경우, 대량의 자료를 읽어 들이게 되면 물리 메모리를 모두 사용하게 되어서 다른 프로그램이 정상 동작하기 힘들게 만드는 문제점이 있었다. 하지만 여기서 제시한 방법을 사용하게 될 경우, 추가적으로 사용하는 메모리의 양이 적어지기 때문에, 다른 프로그램에도 무리가 가지 않게 된다.

추가적으로 현재 읽고 있는 자료 근처의 자료들을 미리 캐싱하여 변수 안에 가지고 있게 된다면, 어느 정도의 성능 향상을 보일 수 있을 것이라 예상한다. 순차적인 자료 접근을 보이거나, 일정 영역에 대하여 잦은 접근을 보이는 연산의 경우, 미리 일정량을 캐싱함으로써 파일에 access 하는 횟수를 줄여 조금 성능이 향상될 수 있다.

3. 토론

비록 기존 시스템 보다는 보다 나은 성능을 보일 수 있는 방법이기는 하지만, 기존 시스템에서 swap file을 사용한 것과 같이 여기서도 disk에 자료를 넣어 놓고 쓰는 방법을 사용했기 때문에 여러 가지 문제점이 있을 수 있다.

먼저, 처음에 자료를 읽어 들이는 시간이 문제점이 될 수 있다. 대량의 자료를 한 번씩 읽는데 걸리는 시간과, 그렇게 읽은 자료를 순차적으로 임시 파일에 저장하는 작업을 해야 하기 때문에, 많은 시간이 소요될 것이라고 예상된다. 하지만, 기존에 직접 메모리로 읽어 들이는 방법에 비해서는 어느 정도 나아진 성능을 보일 것이라 예상한다. 또한, 앞에서 언급한 바와 같이 기존의 파일을 바로 활용한다면, 읽어 들이는 시간도 많이 절약할 수 있을 것으로 보인다.

읽어 들인 행렬에 새로운 열이나 행을 추가, 제거 하는 작업도 문제가 될 수 있다. 중간에 새로운 열이나 행을 추가, 제거 하게 될 경우 파일 전체가 변해야 하는 문제점이 있을 수 있다. 따로 형식을 정의하여 추가되거나 제거된 열이나 행을 표시해 주는 방법으로 해결할 수는 있지만, 이 역시 많이 쌓이게 되면 성능 저하를 보일 수 있기 때문에 더 많은 논의가 필요한 부분이다.

캐싱 역시 많은 토론이 필요한 부분이다. 일반적으로 행렬에서 자료가 읽히는 순서에 따라 효과적인 캐싱 방법이 달라질 수 있다. 만약 시스템에서 사용하는 물리 메모리가 여유가 있다면, 보다 큰 캐시를 사용하여 나은 효과를 노릴 수도 있다. 다양한 캐싱 방법과 캐싱 크기를 가지고 실험을 통해 최적의 방법과 크기를 찾아내는 일이 중요하다.

4. 결론

기존의 R이 가지고 있던 문제점을 해결하고자, 새로운 데이터 형식을 정의하고 임시 파일을 사용하는 방법을 통해서 대량의 자료를 읽어 들이는 방법을 제안하고 그 방법에 대한 여러 가지 이슈들을 서술하였다.

임시 파일을 통해 대량의 자료를 읽어 들이게 되면 기존에 R이 가지고 있던 자료를 읽는데 오랜 시간이 걸린 문제와 읽을 수 있는 자료 크기에 제한이 있었던 문제를 해결할 수 있게 된다. 또한 캐싱이나 기존의 자료 파일을 그대로 사용하는 방법을 써서 어느 정도의 성능 향상도 고려할 수 있다.

참고문헌

[1] The R Project For Statistical Computing, http://www.r-project.org

[2] Journal of Statistical Software, http://www.jstatsoft.org

[3] Maechler M. Sparse Matrices for Large Data Modeling. useR! 2007

[4] R Internals. The R Manuals. http://cran.r-project.org/doc/manuals/R-ints.pdf