CUDA를 이용한 조합 전수조사 알고리즘의 속도 개선 방법

김영민

국방과학연구소

An Enhancement Method of Algorithms Visiting all Combinations by a CUDA Method

Young-min Kim

Agency for Defence Development

E-mail : ymkim1019@add.re.kr

요 약

n개의 원소로 이루어진 집합 S에서 r개의 원소를 선택하여 만들 수 있는 모든 조합을 평가하여 최적의 조합을 찾아내는 것은 많은 공학적 문제를 일반화하여 풀 수 있는 방법이다. 조합 전수조사 알고리즘은 경우의 수가 매우 크거나 각 조합을 평가하는데 많은 시간이 소요될 경우 알고리즘의 수행 속도가 급격히 저하되는 문제가 있다. 본 논문은 CUDA를 이용하여 각각의 조합을 GPU상의 스레드에서 병렬적으로 평가하는 기법을 제안한다. 실험 결과는 GPU상에서 동작하는 병렬적인 알고리즘이 CPU상에서 동작하는 순차적인 알고리즘에 비해 최대 약 900배의 성능 향상이 있음을 보인다.

ABSTRACT

Visiting k-combinations of a set S which has n elements is the general representation of many engineering problems. The performance of algorithms visiting all combinations, however, dramatically degrades with growing cases and the time to evaluate each combination. This paper presents the method to enhance the performance of these algorithms by a CUDA method. The experimental results show that the parallel algorithm running on GPU is approximately 900 times faster than the serial algorithm running on CPU.

키워드

CUDA, 병렬처리, 조합, 전수조사

Ⅰ. 서 론

모든 가능한 해 중 최적의 해를 찾는 것은 문제 해결을 위한 가장 중요한 단계 중 하나이다. 경우의 수가 매우 클 경우 최적의 해가 갖는 특징을 이용하여 평가의 대상을 감소시키는 것은 빠른 시간 내에 최적의 해를 도출하기 위한 필수 조건이다. 그러나 복잡한 문제 또는 최적의 해가 갖는 성질이 알려지지 않은 경우에는 최적의 해를 구하기 위해 모든 경우의 수를 조사해야 한다. 여러 종류의 전수조사 알고리즘 중 n개의 원소로 이루어진 집합에서 r개의 원소를 선택하여 만들 수 있는 모든 조합을 생성하고 이를 평가하는 것은 많은 공학 문제의 일반화된 표현이다. 그러나 조합 전수조사 알고리즘은 경우의 수 증가에 따라 급격히 수행 속도가 떨어지는 단점을 가지고 있다. 본 논문은 CUDA 기법을 이용하여 GPU상의 스레드에서 각 조합을 병렬로 생성하고 평가하는 기법을 제안한다. 실험 결과는 GPU상에서 동작하는 병렬화 된 알고리즘이 CPU상에서 동작하는 순차적인 알고리즘에 비해 최대 약 900배의 성능 향상이 있음을 보인다.

II. 관련 연구

주어진 원소를 이용하여 조합을 생성하는 가장 기본적인 방법은 반복문을 사용하는 것이다. n개의 원소를 가진 집합 S에서 r개의 원소를 조합하는 것은 r개의 반복문을 중첩하여 사용함으로써 쉽게 구현이 가능하다. 그림 1은 10개의 원소가 1부터 10까지 번호가 매겨져있다고 가정했을 때, 3개의 원소를 조합하는 경우를 반복문으로 구현한 코드를 나타낸다. 반복문을 이용한 조합 전수조사 기법은 구현이 간단하다는 장점이 있지만 r의 값에 따라 반복문의 중첩도가 달라지기 때문에 고정된 r에 대해서만 조합 전수조사가 가능하다는 단점이 있다.

```
for (i = 10 ; i > 0 ; i--)
{
    for (j = i - 1 ; j > 0 ; j--)
    {
        for (k = j - 1 ; k > 0 ; k--)
        {
            // 생성된 조합을 평가
        {
    }
}
```

그림 1. 반복문을 이용한 조합 전수조사 기법

GNU Scientific Library (GSL)[1]은 GPL 라이선스를 따르는 공개 수치 라이브러리이다. GSL은 주어진 n과 r의 값에 따라 모든 조합을 순차적으로 생성하는 함수를 제공한다. 그림 2는 GSL을 이용하여 그림 1과 같은 경우의 조합을 생성하는 코드를 나타낸다.

```
gsl_combination* c;
c = gsl_combination_calloc(10, 3);

do
{
    // 생성된 조합을 평가
}
while(gsl_combination_next(c)==GSL_SUCCESS);

gsl_combination_free(c);
```

그림 2. GSL을 이용한 조합 전수조사 기법

Donald Knuth는 [2]를 통해 여러 종류의 프로그래밍 알고리즘과 이에 대한 분석을 기술했는데 이 중 조합을 생성하는 알고리즘 또한 그림 3과 같이 소개되어 있다.

GSL과 Knuth의 알고리즘은 반복문을 이용한 알고리즘에 비해 임의의 n과 r에 대해 조합을 생성할 수 있다는 장점이 있으나 조합을 순차적으로 생성하고 평가한다는 점에서는 반복문을 이용한 알고리즘과 동일하다.

```
            int i, j=1, *c, x;
            int n = 10;
            int k = 3;
            c = malloc( (k+3) * sizeof(int));

            tic();

            for (i=1; i <= k; i++) c[i] = i;
            c[k+1] = n+1;
            c[k+2] = 0;
            j = k;
visit:
            // 생성된 조합을 평가

            if (j > 0) {x = j+1; goto incr;}
            if (c[1] + 1 < c[2])
            {
             c[1] += 1;
             goto visit;
            }
            j = 2;
do_more:
            c[j-1] = j-1;
            x = c[j] + 1;
            if (x == c[j+1]) {j++; goto do_more;}
            if (j > k) exit(0);
incr:
            c[j] = x;
            j--;
            goto visit;
```

그림 3. Knuth의 조합 전수조사 알고리즘

III. CUDA 기반의 조합 전수조사 알고리즘

순차적으로 조합을 생성 및 평가하는 알고리즘은 n과 r의 값이 크거나 하나의 조합을 평가하는데 걸리는 시간이 긴 경우에 알고리즘의 수행시간이 급격하게 증가한다. 이러한 문제를 해결하기 위해 본 논문에서는 CUDA[3]에서 제공하는 병렬 프로그래밍 라이브러리를 이용하여 조합을 GPU의 코어에서 동작하는 각각의 스레드를 이용하여 생성하고 평가하는 기법을 제안한다.

알고리즘을 병렬화하기 위해서는 각각의 스레드에 할당된 코드가 서로 독립적으로 동작해야 한다. 앞에서 소개된 조합 전수 조사 알고리즘은 새로운 조합의 생성이 이전에 생성된 조합의 상태에 기반을 두기 때문에 병렬처리 기법을 적용하는데 어려움이 있다. 조합 생성을 병렬화하기 위해 본 논문에서는 n과 k가 주어졌을 때 생성할 수 있는 조합이 1부터 ${}_nC_k$까지 번호가 매겨져 있다고 가정한다. 또한 집합의 원소에는 1부터 n까지의 번호가 부여되어 있다. 각 스레드는 생성해야 하는 조합의 번호만을 넘겨받으며 전달받은 번호에 해당하는 조합의 생성 및 평가는 다른 스레드와 관계없이 수행된다. 그림 4는 n, k, 조합의 번호가 주어졌을 때 조합을 생성하고 평가하는 알고리즘을 나타낸다.

```
int n = 10, r = 3, idx = 1000;
int i;
int combination[MAX_N]; // 생성된 조합을 저장
int count = 0; // 결정된 원소의 수
int pos = 1; // 현재 생성된 조합의 번호
int num_of_combinations;

for (i = n ; i > 0 ; i--)
{
   // 마지막 원소를 결정하는 경우
   if ( (r - count) == 1)
   {
      combination[count] = i - (idx - pos);
      count++;
      break;
   }
   // 마지막 원소가 아닌 경우
   else
   {
      /* 현재 위치의 원소를 i로 결정했을 때
         만들 수 있는 조합의 수 */
      num_of_combinations
       = number_of_combinations(i-1, r-count-1);

      /* 주어진 조합의 번호가 현재 위치의 원소를
         i로 정하여 생성할 수 있는 조합의 범위
         안에 들어올 경우 */
      if (idx <= (pos + num_of_combinations-1) )
      {
          // 현재 위치의 원소를 i로 결정
          combination[count] = i;
          count++;
      }
      else
      {
          pos += num_of_combinations;
      }
   }
}

// 생성된 조합을 평가
```

그림 4. 병렬화된 조합 전수조사 기법

알고리즘은 조합을 구성하는 원소를 순차적으로 구한다. 예를 들어 그림 4에 주어진 것과 같이 10개의 원소 중 3개의 원소를 조합하여 만들 수 있는 조합의 수는 $_{10}C_3$ = 120개이다. 이 중 가장 큰 원소들로 구성된 조합부터 내림차순으로 10번째 조합을 구한다고 가정하자. 먼저 첫 번째 원소를 10이라 가정하면 나머지 2개의 원소를 선택하는 경우의 수는 $_9C_2$ = 36개이다. 첫 번째 원소를 10으로 정했을 때 만들 수 있는 조합은 1부터 36까지의 번호를 갖는다. 생성하고자 하는 조합의 번호 10이 해당 범위 안에 위치하기 때문에 첫 번째 원소를 10으로 정할 수 있다. 이와 같이 나머지 원소를 결정함으로써 10번째로 위치한 조합을 생성할 수 있다.

병렬적인 알고리즘은 조합 한 개의 생성에 있어서는 순차적인 알고리즘보다 긴 시간을 필요로 하지만 각 스레드에서 병렬적으로 조합의 생성 및 평가가 가능하기 때문에 모든 경우의 수를 평가하는데 에는 더 적합하다.

Ⅳ. 실험 결과

제안된 알고리즘의 성능을 기존 알고리즘들과 비교하기 위해 표 1과 같은 환경에서 실험을 수행하였다.

표 1. 실험 수행 환경

항목	세부 환경
CPU	Intel i7-3770
RAM	4G
GPU	GTX650
CUDA	SDK v5.0 GRID_DIM = 32 BLOCK_DIM = 256

모든 알고리즘에 대해 100개의 원소 중 4개의 원소를 선택하여 만들 수 있는 3,921,225개의 조합을 생성하고 평가하는데 걸리는 시간을 측정하였으며, 각 조합의 평가에 걸리는 시간이 전체 수행 시간에 미치는 영향을 알아보기 위해 각 조합의 평가하는 연산을 반복문의 1000에서 3000번까지의 단순 수행으로 모의하였다.

표 2. 알고리즘 별 조합 전수조사 수행 시간

알고리즘	조합평가 방법(반복문 수행 횟수)에 따른 전수 조사 시간				
	1000	1500	2000	2500	3000
반복문	6.594	9.86	13.156	16.406	19.672
GSL	7.125	10.39	14.047	17.203	20.484
Knuth 알고리즘	6.625	9.922	13.218	16.578	19.844
병렬처리 알고리즘	0.017	0.017	0.017	0.0188	0.0218

표 2는 각 알고리즘의 조합 전수조사 수행 시간을 나타낸다. 수행 시간의 단위는 초이며 표에서 확인할 수 있듯이 병렬처리 기법을 적용한 알고리즘이 순차적인 알고리즘에 비해 월등히 빠른 수행시간을 보인다. 또한 조합을 평가하는데 걸리는 시간이 클수록 병렬처리의 이점이 큰 것을 알 수 있다. 조합을 평가하는데 반복문 1000번만큼의 시간이 걸린다고 가정했을 때의 성능향상은 약 400배인 것에 반해 조합 평가 시간을 반복문 3000번 수행으로 가정했을 때의 성능향상은 약 900배 이상에 달한다. 이는 수행 시간이 긴 부분을 우선적으로 병렬화하는 병렬처리 기법의 기본과 일치하는 결과이다.

Ⅴ. 결 론

본 논문은 다양한 문제를 일반화하여 해결할 수 있는 조합 전수조사 알고리즘의 성능을 높이기 위한 병렬화 기법을 제안하였다. 조합의 생성 및 평가를 GPU 코어에서 동작하는 각 스레드가 병렬적으로 처리함으로서 순차적인 조합 전수조사 알고리즘에 비해 주목할 만한 성능 향상을 이끌어낼 수 있었다. 앞으로 제안된 알고리즘을 다양한 실제의 문제에 적용하여 병렬화 기법의 효과를 검증할 예정이다.

참고문헌

[1] “GNU Scientific Library”, www.gnu.org/software/gsl/

[2] Knuth, Donald E. Art of Computer Programming, Volume 4, Fascicle 4, The: Generating All Trees-History of Combinatorial Generation. Addison-Wesley Professional, 2006.

[3] “CUDA”, www.nvidia.com/object/cuda_home_new.html