어셈블리프로그램 설계 및 실습

Term project 제안서

제출일자: 2021년 11월 21일 (일)

학 과: 컴퓨터정보공학부

담당교수: 이준환 교수님

학 번: 2018202060

성 명: 이 준 형

1. 과제제목

Matrix convolution

행렬의 합성곱

1. 과제목표

임의의 부동소수점 데이터로 이루어진 2N-by-2N 행렬에 padding을 수행한 뒤 convolution 및 sub-sampling을 수행한 N-by-N의 행렬을 출력하는 어셈블리 코드를 작성할 수 있다.

1. 일정

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 기간  항목 | 1주차 |  |  |  |  |  |  | 2주차 |
| 11월 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 |
| 제안서 제출 |  |  |  |  |  |  |  |  |
| 문제정의 및 정보수집 |  |  |  |  |  |  |  |  |
| Coding |  |  |  |  |  |  |  |  |
| coding 검증 |  |  |  |  |  |  |  |  |
| Report 작성 |  |  |  |  |  |  |  |  |
| 최종 제출 |  |  |  |  |  |  |  |  |

1. 각 function 별 알고리즘
2. Padding

Input : 2n-by-2n 행렬을 Input으로 넣는다.

이 Input이 입력된 2n \* 2n 의 크기를 가지는 행렬의 가장자리(boundary)의 경우 위치가 가장 가까운 값을 복사하여 Output : 2n + (k-1) - by -2n + (k-1) 행렬이 나온다. 만약에 n=2, k=3이라고 가정을 하고, 행렬의 원소를 숫자로 표현을 한다면 아래와 같은 6\*6구조를 가진다.

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |

Matrix\_first

그리고 주어진 operation을 수행한다면, 노란색과 같은 padding이 추가가 된다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 1 | 2 | 3 | 4 | 4 |
| 1 | 1 | 2 | 3 | 4 | 4 |
| 5 | 5 | 6 | 7 | 8 | 8 |
| 9 | 9 | 10 | 11 | 12 | 12 |
| 13 | 13 | 14 | 15 | 16 | 16 |
| 13 | 13 | 14 | 15 | 16 | 16 |

padded\_Matrix\_first

1. Convolution

Input은 다음과 같다. 2n + (k-1) - by -2n + (k-1), 그리고 이 행렬을 곱해줄 k-by-k행렬. 이 input으로 이 Input은 k-by-k 크기의 행렬을 행렬곱을 계산하여 나온 행렬 2n-by-2n행렬을 output으로 한다. 위 A와 같은 n(=2)과 k(=3)를 가진다면 padding이 된 matrix인 6-by-6행렬과 3-by-3를 곱해서 2n-by-2n의 행렬을 만들면 된다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 1 | 2 | 3 | 4 | 4 |
| 1 | 1 | 2 | 3 | 4 | 4 |
| 5 | 5 | 6 | 7 | 8 | 8 |
| 9 | 9 | 10 | 11 | 12 | 12 |
| 13 | 13 | 14 | 16 | 16 | 16 |
| 13 | 13 | 14 | 15 | 16 | 16 |

|  |  |  |
| --- | --- | --- |
| A | B | C |
| D | E | F |
| G | H | I |

|  |  |  |  |
| --- | --- | --- | --- |
| R1 | R2 | R3 | R4 |
| R5 | R6 | R7 | R8 |
| R9 | R10 | R11 | R12 |
| R13 | R14 | R15 | R16 |

\* =

Matrix\_padding matrix\_second matrix\_convolution

Matrix\_second의 중심(e)이 위치한 좌표가 Matrix\_first의 (0,0)인 경우, 해당 연산결과는 Matrix\_convolution의 (0,0)에 위치하게 된다.

위와 같은 예시인 경우에 3x3행렬을의 연산곱을 16번해야해서 states수가 많을것으로 예상된다.

1. Sub-sampling

Input은 다음과 같다. 2n-by-2n행렬. 이 Input은 1/2 크기로 sub-sampling하기 위해서 짝수행과 짝수열의 데이터만을 선택하여, 선택된 데이터를 sub-sampling을 한다. 아래는 예시이고(padding과정과 convolution과정처럼 n=2인 상황이다.)

|  |  |  |  |
| --- | --- | --- | --- |
| 0, 0 | 0, 1 | 0, 2 | 0, 3 |
| 1, 0 | 1, 1 | 1, 2 | 1, 3 |
| 2, 0 | 2, 1 | 2, 2 | 2, 3 |
| 3, 0 | 3, 1 | 3, 2 | 3, 3 |

|  |  |
| --- | --- |
| 0, 0 | 0, 2 |
| 2, 0 | 2, 2 |

sub-sampling

======>

matrix\_covolution Matrix\_result

이번 프로젝트 같은 경우에는 n=32이고, k=3이다.

padding과정의 input matrix는 64-by-64이고, output은 66-by-66이다.

convolution과정의 input matrix는 66-by-66, 3-by-3이고, output은 64-by-64이다.

sub-sampling과정의 input matrix는 64-by-64이고, output은 32-by-32이다.

1. 예상되는 문제점

66-by-66행렬을 다 저장하는것은 states수가 기하급수적으로 늘어날것으로 예상되어서 padding만 따로 save하는게 좋아보인다.

Multiplication에서 mantissa값을 각 24bit와 24bit를 곱해서 결과값 48bit가 나오게 된다. 하지만 어셈블리어에서 사용하는 register은 32bit이다. 그러므로 result 48bit의 값을 두개의 register에 나눠서 저장할 수 있어야한다.

66x66의 행렬을 저장할 때 겹치는 부분을 잘 파악하여 없애고 행과 열의 순서를 맞춰서 저장할 수 있어야 한다.

어떤 식으로 padding이 되는지 잘 파악하고 Adder와 Multiplication의 순서를 잘 파악한 후, 연산을 진행하여, 다음 함수에 input으로 전달하도록 해야한다.

많은 양의 데이터를 연산하고 로드하고 세이브 해야해서 state를 줄이는 것이 문제가 될것이다.

1. 검증전략

1. 맨먼저 값들로 행렬을 구성하여 나오는지 확인한다.

2. padding이 되고나서 어떻게 값이 저장했는지 확인한다.

3. 그 후, matrix\_second의 값들이 행렬을 구성하여 나오는지 확인한다.

4. 행렬의 convolution을 진행한 후, 값이 제대로 계산되었는지 확인한다.

5. 그 후, sub-sampling과정을 시행한 후, 짝수행과 짝수열의 data가 제대로 저장되었는지 확인한다.