Parallel Programming

Amdahl's law.

$$Speedup_{MAX} = \frac{1}{(1-P) + \frac{P}{N}}$$

P... portion of being P
N... # of processors

Gustafson's law

Speedup = P - a(n) · (

P... # of processor
a(n)... portion of being

암달의 법칙과 구스타프슨의 법칙

애들아빠 @
http://macosala.blogspot.com
http://groups.google.com/group/lisp-korea

본 자료는 wikipedia 에 있는 내용들을
바탕으로 정리한 것입니다.

암달의 법칙이란?

시스템의 일부가 개선될 때
전체 시스템에서 어느 정도의
성능 개선이 기대될 수 있는지를
나타내는 데 사용된다

$$Speedup_{MAX} = \frac{1}{(1-p)+\frac{p}{S}}$$

p : 시스템 전체 대비 개선된 부분
S : 속도 향상이 일어난 정도

암달의 법칙이란?

시스템의 일부가 개선될 때
전체 시스템에서 어느 정도의
성능 개선이 기대될 수 있는지를
나타내는 데 사용된다

$$Speedup_{MAX} = \frac{1}{(1-p) + \frac{p}{S}}$$

p : 시스템 전체 대비 개선된 부분
S : 속도 향상이 일어난 정도

예를 들어, 전체 시스템의 30% 부분에서
2배의 속도 향상이 있었다면...

암달의 법칙이란?

시스템의 일부가 개선될 때
전체 시스템에서 어느 정도의
성능 개선이 기대될 수 있는지를
나타내는 데 사용된다

$$Speedup_{MAX} = \frac{1}{(1-p)+\frac{p}{S}}$$

p : 시스템 전체 대비 개선된 부분
S : 속도 향상이 일어난 정도

예를 들어, 전체 시스템의 30% 부분에서
2배의 속도 향상이 있었다면...

$$Speedup_{MAX} = \frac{1}{(1-0.3)+\frac{0.3}{2}} = 0.17647$$

암달의 법칙이란?

시스템의 일부가 개선될 때
전체 시스템에서 어느 정도의
성능 개선이 기대될 수 있는지를
나타내는 데 사용된다

$$Speedup_{MAX} = \frac{1}{(1-p)+\frac{p}{S}}$$

p : 시스템 전체 대비 개선된 부분
S : 속도 향상이 일어난 정도

예를 들어, 전체 시스템의 30% 부분에서
2배의 속도 향상이 있었다면...

$$Speedup_{MAX} = \frac{1}{(1-0.3)+\frac{0.3}{2}} = 0.17647$$

전체 시스템에서는 약 17.6%의 성능 향상이 발생한다

이를 병렬처리 부문으로 적용하면...

$$Speedup_{MAX} = \frac{1}{(1-p)+\frac{p}{N}}$$

p : 전체 시스템 중 병렬 처리 가능한 부분(0~1)
N : 프로세서(또는 코어)의 개수

이를 병렬처리 부문으로 적용하면...

$$Speedup_{MAX} = \frac{1}{(1-p) + \frac{p}{N}}$$

그런데, 이론적으로 코어를 무한정으로 확장 가능하다 가정하면,

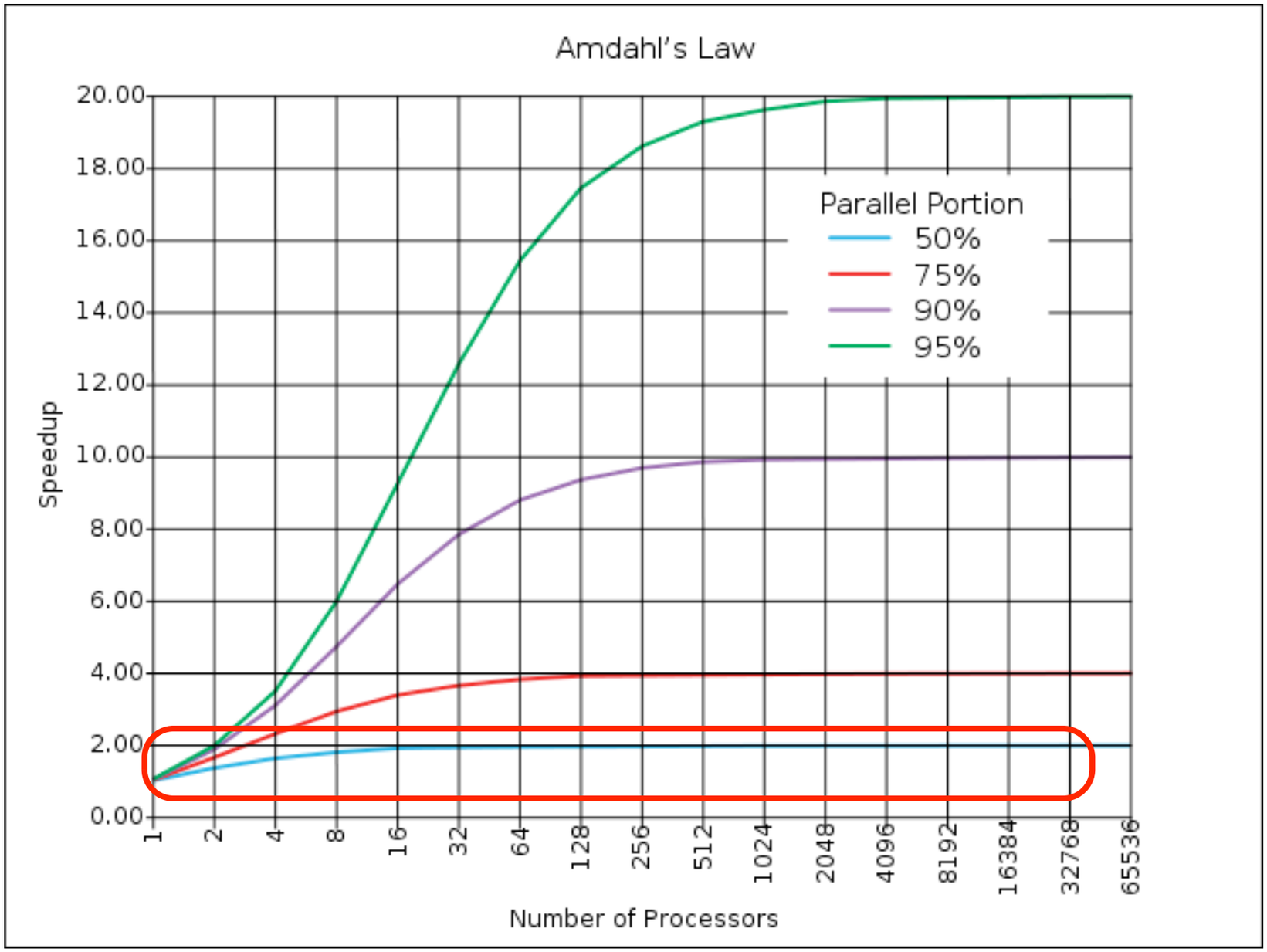
$\lim_{N \to \infty} \frac{p}{N} = 0$ 이 되므로,

$Speedup_{MAX} = \frac{1}{(1-p)}$ 과 같이 된다.

암달의 법칙에 따른

병렬처리의 효과

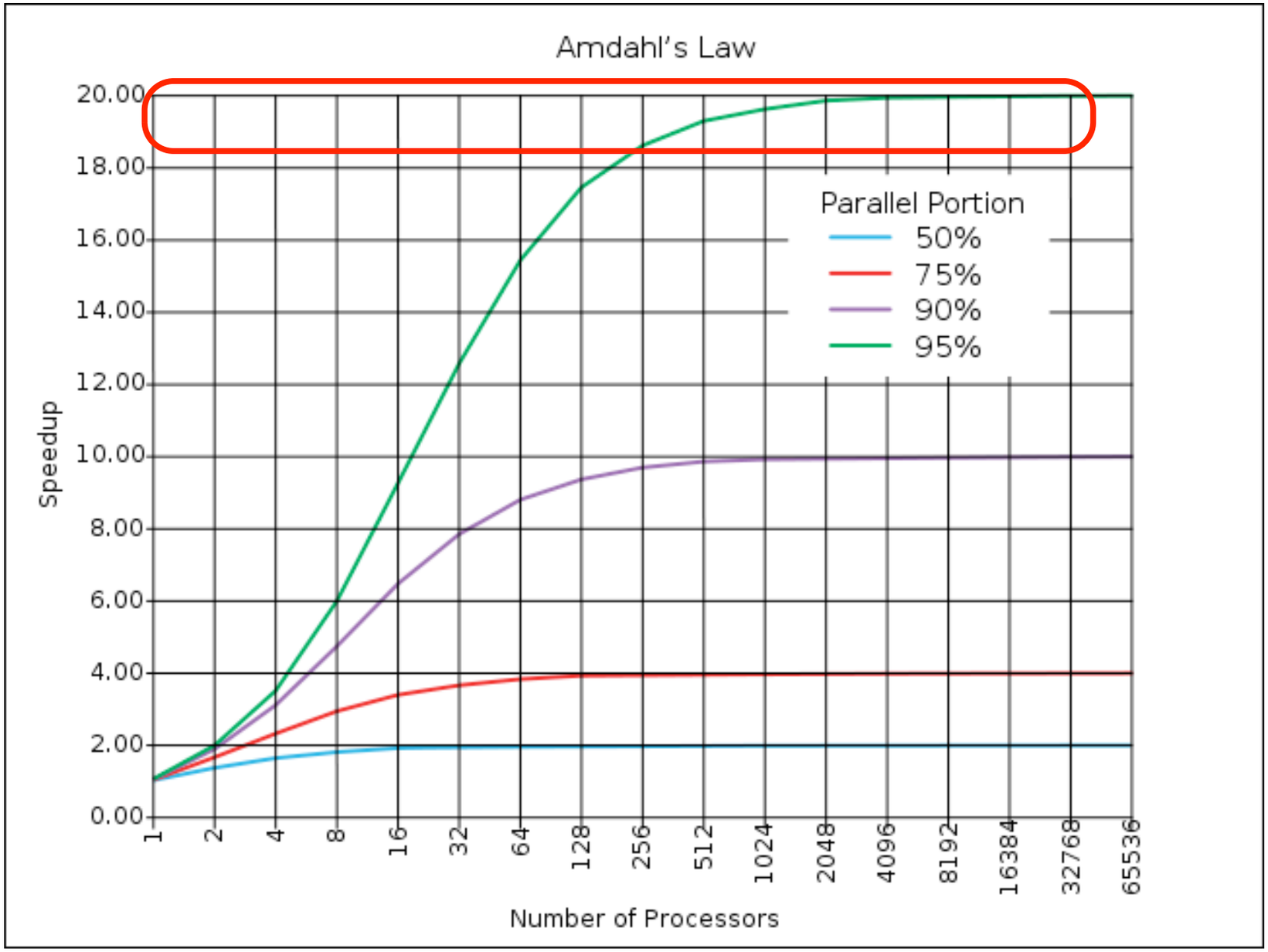
암달의 법칙에 따르면 병렬화 부분이 50%인 경우에는 전체 시스템의 성능 향상이 2배에서 수렴하며,



암달의 법칙에 따른

병렬처리의 효과

병렬화율이 95%에 달하는 경우라도 20배 정도에서 성능 향상이 수렴된다.



암달의 저주 ?!

암달의 법칙에 따르면 아무리 코어를 많이 확충한다 하더라도 시스템의 성능 향상에는 한계가 있다.

그러므로, 병렬화될 수 있는 부분을 최대화하는 노력과 함께 비교적 소규모 프로세서에서 시스템을 돌리는 것을 고려하여야 한다.

이에 대한 John Gustafson 의 반박

대규모의 반복적인 데이터들을 대상으로 하는 문제들은 효율적으로 병렬화될 수 있다는 법칙

문제를 충분히 작게 만들어서 현재 사용가능한 프로세서가 그 문제를 실질적인 시간 안에 풀 수 있도록 하면, 나중에 더 빠른(더 많은)프로세서를 사용해서 더 큰 문제도 같은 시간 내에 해결할 수 있다고 주장

"우리는 전산학에 몸담고 있는 사람들이 암달의 성능개선 공식을 잘못 사용함으로써 생기는 '정신적 한계점'을 넘어서야 한다고 생각한다.

성능 개선치 문제를 프로세서의 개수로 스케일링 하여 측정되어야 하며, 문제 크기를 고정시킨 상태에서 측정되면 안된다."

http://www.scl.ameslab.gov/Publications/Gus/AmdahlsLaw/Amdahls.html

Gustafson 의 법칙이란?

n을 문제의 크기라고 할 때, 병렬화되지 않은 부분을 a(n), 병렬화된 부분을 b(n)이라 할 때, 병렬 환경에서의 프로그램의 실행은 아래와 같이 기술된다.

$$a(n) + b(n) = 1$$

위 경우를 완전히 순차적인 환경에서 돌린다고 한다면, 상대적인 실행 시간은 다음과 같다. (p는 병렬 환경에서 프로세서의 개수이다.)

$$(a(n) + p \cdot b(n))$$

그러므로, 성능 개선치(speedup)는 다음과 같이 표현된다

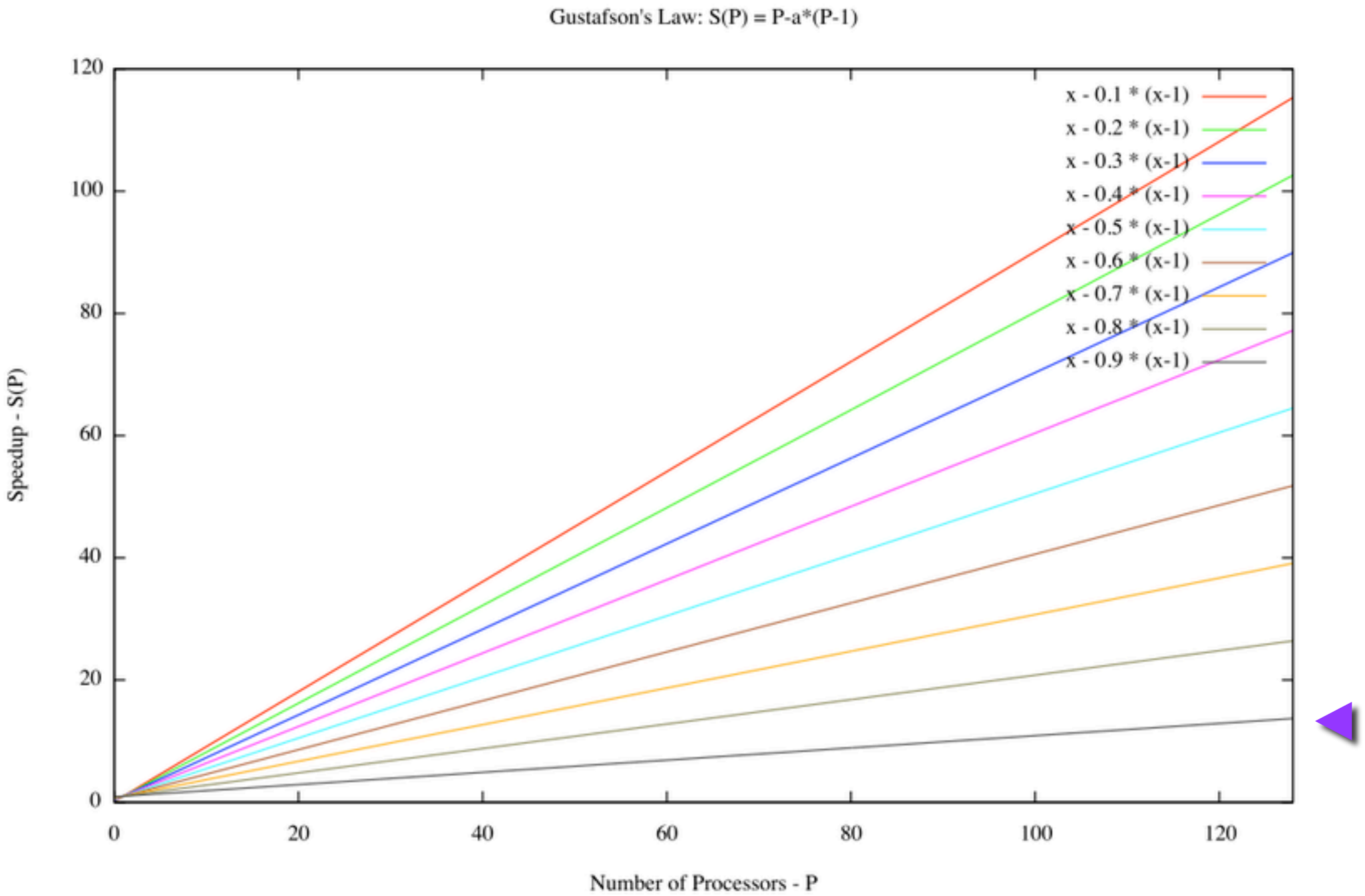
$$S = a(n) + p \cdot (1 - a(n)) \quad \text{또는,}$$

$$S = p - a(n) \cdot (1 - p)$$

Gustafson 의 법칙에 따른

병렬처리의 효과

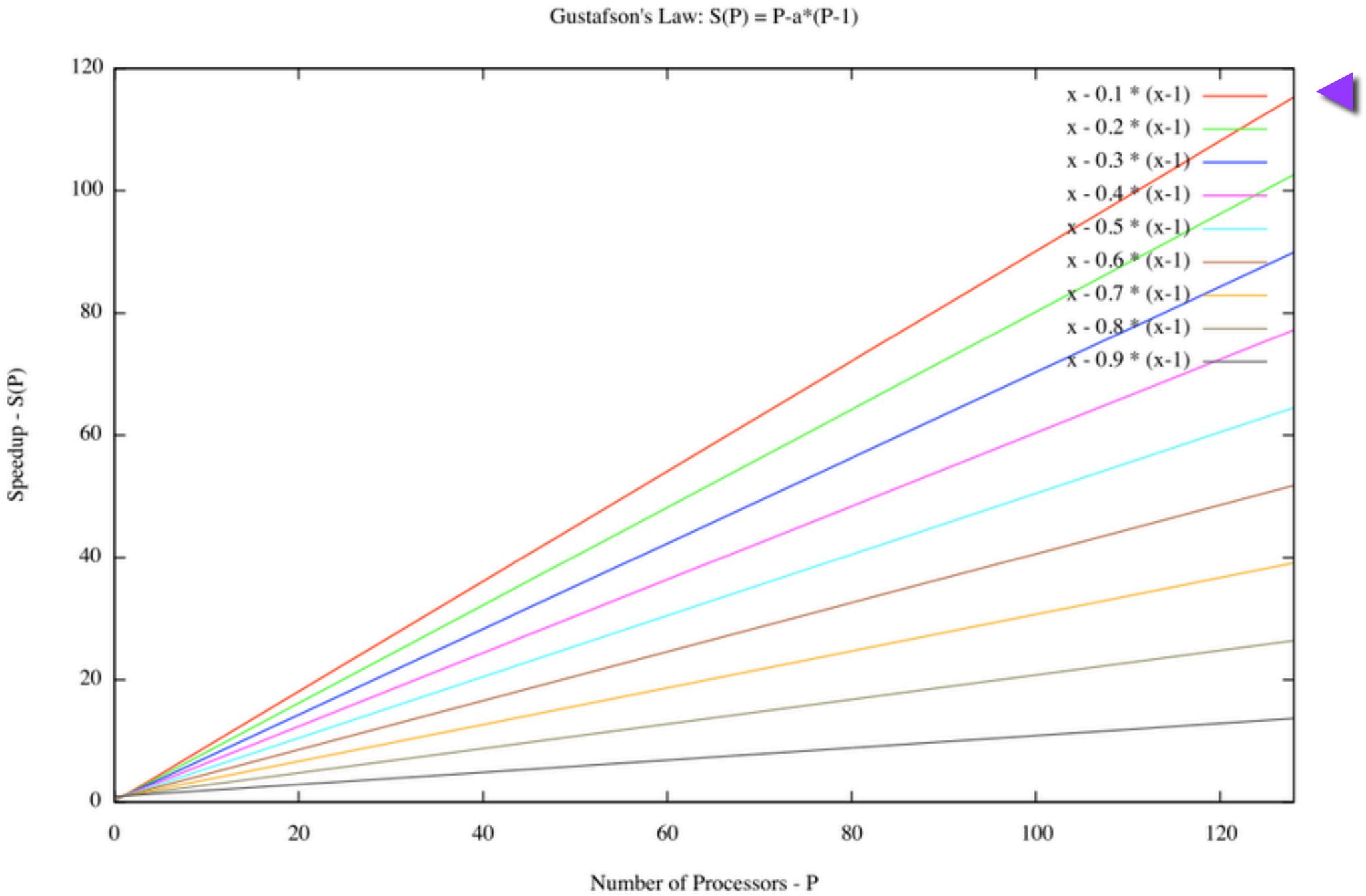
병렬화율이 10%인 경우에도 프로세서의 개수가 증가하면서 처리속도가 향상되며,



Gustafson 의 법칙에 따른

병렬처리의 효과

병렬화율이 90%인 경우에는 거의 선형적인 속도 증가를 예측할 수 있으며, 암달의 예측과는 달리 거의 수렴하지 않는다.



Scalable parallelism

더 큰 문제들을 푸는 데 추가적인 프로세서 자원을 사용하는
소프트웨어를 “Scalable Parallelism” 을 보인다고 하는데,
이는 Gustafson’s Law 에 합당하다는 것을 의미한다.

Scalable parallelism

더 큰 문제들을 푸는 데 추가적인 프로세서 자원을 사용하는
소프트웨어를 “Scalable Parallelism” 을 보인다고 하는데,
이는 Gustafson’s Law 에 합당하다는 것을 의미한다.

```
for t := 0 to T do
    for i := 1 to N-1 do
        new(i) := (A(i-1) + A(i) + A(i) + A(i+1)) * .25
        // explicit forward-difference with R = 0.25
    end
    for i := 1 to N-1 do
        A(i) := new(i)
    end
end
```

finite difference heat equation stencil calculation

위와 같은 프로그램은 t 루프 안에 있는 각각의 i 루프를 병렬
형태로 돌릴 수 있으므로, 프로세서가 증가하면서 수행속도도
함께 증가하게 된다.

그러나...

Gustafson 법칙의 한계

문제를 풀기 위해 대용량의 데이터 셋을 필요로 하지 않는 경우도 있는데, Gustafson 법칙에 따르면 그러한 경우에는 병렬화의 장점을 살리기 어렵다

Snyder는 비선형계획법 알고리즘들의 경우 Gustafson 법칙에서 주장하는 병렬화의 장점을 가지기 힘들다. $O(N^3)$ 인 경우에는 프로세서를 두 배로 늘려도 26% 정도의 문제 크기 증가만이 가능하다고 주장하였다.

Type Architectures, Shared Memory, and The Corrolary of Modest Potential