다중 퍼지 정책에 의한 페이지 교환 알고리즘

이광형 이도헌 이건명 송운선
한국과학기술원 전산학과

Page Replacement Algorithm by Multiple Fuzzy Strategies

H Lee-Kwang D Lee K Lee Y Song
Dept of Computer Science, KAIST

요약

본 논문에서는 가상기억시스템(virtual memory system)의 페이지 교환 알고리즘에 다중 퍼지 정책을 이용하는 방법을 제안했다 페이지 교환 알고리즘에서 희생자(victim)를 선정하는 문제는 미래의 페이지 요구(page request)상황에 대한 예측을 필요로 하기 때문에 최적의 성능을 보장하는 알고리즘을 설계하기가 곤란하다 따라서 경험적으로 관찰할 수 있는 발견적(heuristic)인 정책을 퍼지규칙(fuzzy rule)의 형태로 표현하고 그것을 효과적으로 적용시키는 방법을 제시했다

1 서론

가상기억시스템(virtual memory system)은 실제적인(physical) 기억용량보다 더 큰 용량의 논리적인(logical) 기억장소를 이용할 수 있도록 지원해주는 기억장치(memory) 관리시스템이다[7] 가상기억시스템을 이용함으로써 실제적인 기억 용량보다 큰 크기의 프로그램을 적재하여 동작시킬 수 있고, 또한 다중프로그래밍 차수(degree of multiprogramming)를 높이게 되어, 전체적인 시스템의 성능을 향상시킬 수 있다

이러한 가상기억시스템을 운영하는 데는 페이지 요구(page request)에 따른 페이지 교환 알고리즘(page replacement algorithm)이 필요하다[7] 즉, 현재 주기억장치에 존재하지 않은 페이지에 대한 요구(request)가 있을 때, 주기억장치에 적재되어 있는 페이지 중에서 적당한 희생자(victim)를 결정하여 디스크로 옮기고 그 자리에, 요구된 페이지를 적재하는 작업이 필요한 것이다 이때 희생자(victim)를 어떻게 선정하느냐에 따라 시스템의 성능이 크게 영향을 받는다[7]

그런데 페이지에 대한 요구가 어떻게 발생할 지는, 그 프로세스를 동작시키기 전에 미리 알 수 없는 일이기 때문에 희생자(victim) 선정에 대한 최적의 알고리즘을 정확히 얻어내는 일이 매우 어려운 일이다

퍼지이론은 발견적인(heuristic) 지식을 표현하는데 적합한 것으로 알려져있다[2] 특히, 시스템의 동작이 매우 복잡하고 환경에 따른 변화요인이 많아서 알고리즘에 의한 분석적인 처리가 곤란할 때, 효과적으로 이용할 수 있다

이에 본 논문에서는 발견적인 지식을 퍼지규칙(fuzzy rule)의 형태로 표현하여 가상기억시스템의 희생자(victim) 선정에 적용하는 방법을 제시하고자 한다

2 기존의 페이지 교환 알고리즘(page replacement algorithm)

가상기억장치의 페이지 교환 알고리즘은 다음과 같이 동작한다[7]

1) 동작중인 프로세스에 의해 특정 페이지 A에 대한 요구(request)가 생긴다
2) 페이지 A가 주기억장치에 존재하면 6)으로 가고, 그렇지 않으면 3)으로 간다
3) 현재 주기억장치에 존재하는 페이지 중에서 적당한 페이지 B를 희생자(victim)로 선정한다
4) 페이지 B를 디스크(disk)로 복사한다
5) 주기억장치에서 페이지 B에 할당되어 있던 부분에 페이지 A를 적재한다
6) 프로세스의 페이지 A에 대한 요구를 처리한다

이때, 2)의 단계에서, 페이지 A가 주기억장치에 존재하고 있지 않은 상황을 페이지 부재(page fault)라고 한다[7]. 그런데 4)와 5)의 단계에서 주기억장치와 디스크간의 페이지 이동이 필요하므로, 페이지 부재가 발생할 경우 그 처리를 위한 시간비용이 많이 소모된다 따라서 가능한 한 페이지 부재가 발생하지 않도록

Copyright (C) 2005 NuriMedia Co., Ltd.

하는 것이 시스템의 성능을 평가하는 관점에서 바람직하다 즉, 희생자(victim)를 선정할 때, 장차 요구(request)가 발생할 가능성이 가장 적은 것을 선택할 수 있는 방법이 요구된다고 하겠다

희생자(victim)를 선정하는 정책(strategy)으로서 최적 페이지 교체(optimal page replacement)정책을 생각할 수 있다 이는 현재 주기억장치에 존재하는 페이지 중에서, 장차 가장 나중에 요구될 페이지를 희생자(victim)로 선정하는 것이다 이 방법은 항상 최적의 성능을 보장한다[7] 그러나 미래의 상황을 정확히 알 수 있어야만 하므로 실제로 구현될 수는 없다 따라서 최적 페이지 교체정책과 근사한(approximate) 효과를 얻을 수 있는 여러 방법이 제안되어 있다

먼저 최소최근사용(LRU least recently used)정책을 들 수 있다 이는 미래에 대한 추측을 위해 과거의 정보를 이용하는 방법이다 즉 가장 긴 시간동안 사용되지 않고 있는 페이지를 앞으로도 사용될 가능성이 가장 적은 페이지로 간주하는 것이다

또한 최소사용빈도(LFU least frequently used)정책을 생각할 수 있다[7] 이는 사용된 횟수가 가장 적은 페이지를 희생자(victim)로 선택하는 것이다 그 역으로 최대사용빈도(MFU most frequently used)정책을 생각할 수 있는데, 이는 최소사용빈도 정책과는 반대로, 사용된 횟수가 가장 많은 페이지가 오히려 앞으로 사용될 가능성이 적다고 간주하는 정책이다[7]

일반적인 스케쥴링 방법중 가장 간단한 선입선출(FIFO fisrt in first out)정책도 희생자(victim) 선정에 사용될 수 있는데, 이것은 벨라디의 이상(Belady's anomaly)이 나타날 수 있어서 그리 효과적인 방법은 아니다

그런데 시스템의 동작 환경에 따라, 위에서 제시한 정책들을 효과적으로 합성하여 적용하면 보다 나은 성능을 기대할 수 있으리라고 생각한다

3 퍼지정책을 이용한 희생자(victim)의 선정

경험적으로 판단했을 때, 페이지의 요구상황이 (그림 4-1)과 같이 나타난다고 가정하자

마지막 사용후 현재까지의 경과된 시간(idle time)이 길면
장차 요구(request)가 발생할 가능성이 적다
현재까지의 사용된 횟수(frequency)가 적으면
장차 요구(request)가 발생할 가능성이 약간 적다
페이지가 적재된 후 현재까지 경과한 시간(resident time)이 길면
장차 요구(request)가 발생할 가능성이 약간 적다

(그림 4-1) 경험적으로 본 페이지 요구 상황

이때, 장차 요구가 발생할 가능성이 적을 수록, 희생자(victim)로 선정될 우선순위가 높다고 볼 수 있으므로 (그림 4-2)와 같이 퍼지규칙(fuzzy rule)[1]의 형태로 바꿀 수 있다

마지막 사용후 현재까지의 경과된 시간(idle time)이 길면
희생자(victim)로서의 우선순위가 높다 LRU 정책
현재까지의 사용된 횟수(frequency)가 적으면
희생자(victim)로서의 우선순위가 약간 높다 LFU 정책
페이지가 적재된 후 현재까지 경과한 시간(resident time)이 길면
희생자(victim)로서의 우선순위가 약간 높다 FIFO 정책

(그림 4-2) 퍼지규칙으로 표현한 희생자 (victim) 선정 정책

물론 (그림 4-1)에서 나타난 상황은 임의로 가정한 것이고, 시스템의 동작환경에 따라 다양한 상황을 부과할 수 있을 것이다

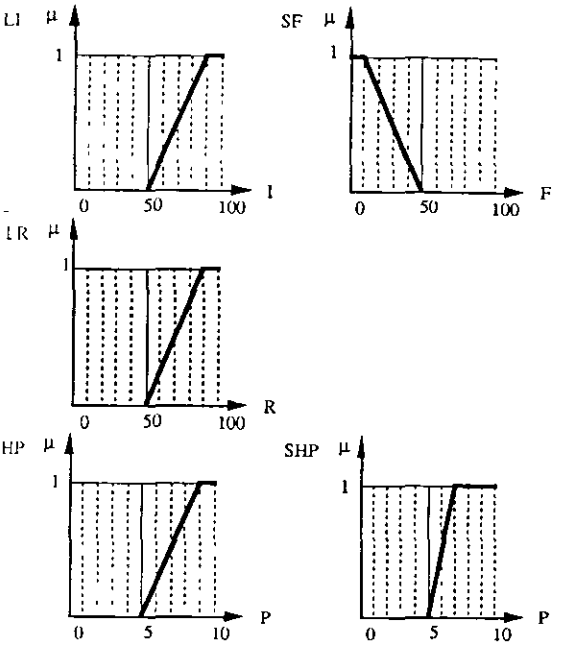
이때, (그림 4-2)에 나타난 퍼지규칙(fuzzy rule)을 처리하기 위하여 다음과 같은 집합들을 정의한다

$I = \{ i \mid 0 \le i \le 100 \}$ 가능한 idle time의 집합,
$F = \{ f \mid 0 \le f \le 100 \}$ 가능한 frequency의 집합,
$R = \{ r \mid 0 \le r \le 100 \}$ 가능한 resident time의 집합,

$P = \{ p \mid 0 \le p \le 10 \}$ 가능한 priority의 집합,

LI (Idle time이) 길다 = $\{ (i, \mu_{LI}(i)) \mid i \in I \}$,
SF (frequency가) 적다 = $\{ (f, \mu_{SF}(f)) \mid f \in F \}$,
LR (Resident time이) 길다 = $\{ (r, \mu_{LR}(r)) \mid r \in R \}$,
HP (Priority가) 높다 = $\{ (p, \mu_{HP}(p)) \mid p \in P \}$,
SHP (Priority가) 약간 높다 = $\{ (p, \mu_{SHP}(p)) \mid p \in P \}$,

단, 여기에서 각각의 소속함는 (그림 4-3)과 같이 정의한다

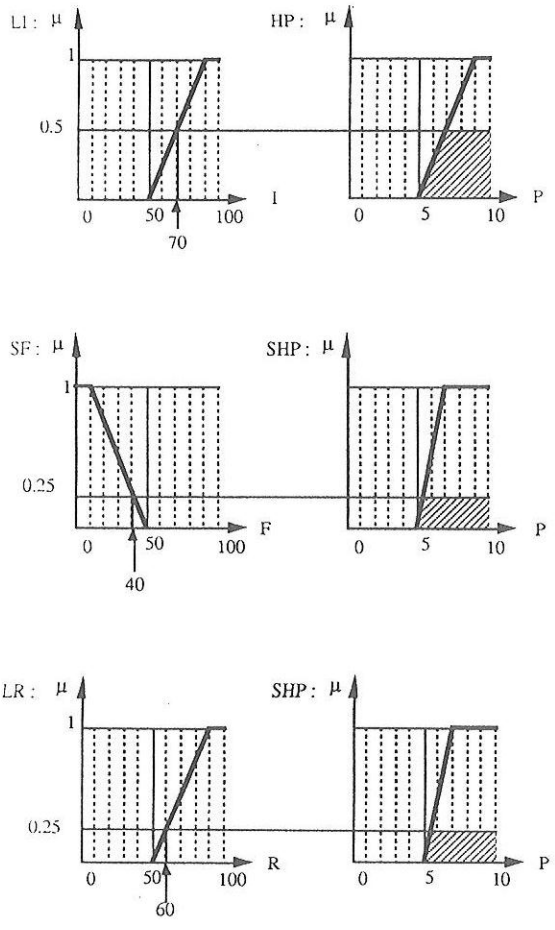


(그림 4-3) 각 퍼지집합의 소속함수

Copyright (C) 2005 NuriMedia Co., Ltd.

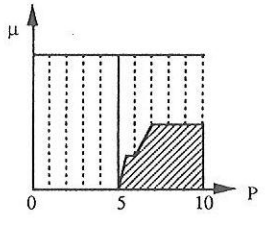
위와 같이 정의하고 (그림 4-2)의 규칙을 보간법(interpolation)[2]을 이용하여 적용해보기로 하자.

현재 주기억장치에 있는 페이지 A의 idle time, frequency, resident time이 각각 70, 40, 60 이고, 페이지 B의 idle time, frequency, resident time이 각각 60, 30, 70 이라고 하자. 그러면 페이지 A의 경우를 보자.



(그림 4-4) 보간법(interpolation)을 이용한 퍼지 규칙의 적용 모습

(그림 4-4)와 같이 각 규칙이 적용되고 각각의 결과에 대한 퍼지 합집합 연산[4]을 수행하면 (그림 4-5)와 같다.



(그림 4-5) 각각의 결과에 대한 합집합 연산

여기서 무게 중심법[2]을 이용하여, 결과로 얻어진 퍼지집합의 대표값을 구하면

페이지 A의 우선순위 = 8.03

이 된다.

같은 방법으로 페이지 B의 우선순위를 구하면

페이지 B의 우선순위 = 7.78

이 된다.

그러므로 페이지 A가 페이지 B보다 우선순위가 높다고 말할 수 있다. 이와 같은 방법으로 주기억장치 내의 페이지들을 비교하여 가장 우선순위가 높은 페이지를 희생자(victim)으로 선정할 수 있다.

4. 결론

기존의 최소최근(LRU)정책, 최소사용빈도(LFU)정책, 최대사용빈도(MFU)정책, 선입선출(FIFO)정책등과는 달리 여러가지 정책을 효과적으로 합성하여 반영할 수 있었다. 3절에서 예를 든 경우에서 페이지 A가 최소최근정책이나 선입선출정책에 의해서는 페이지 B보다 우선순위가 높으나 최소사용빈도정책의 관점에 의해서는 우선순위가 낮은 것을 알 수 있다. 그런 상황에서 기존의 방법은 어떤 특정 정책만을 이용하여 희생자(victim)를 결정해야 했으나 본 논문의 방법을 이용하면 각 정책을 모두 효과적으로 반영할 수 있었다.

또한 시스템의 동작상황에 따라 희생자(victim)의 선정 정책에 변화를 줄 수 있다. 즉, 동작환경에 따라 미래의 페이지 요구(request)를 예측하여 볼 때, idle time보다 frequency가 더욱 중요한 요인으로 작용할 수 있는 것이다. 그런 경우에도 (그림 4-2)와 같은 형태의 규칙을 변경함으로써 효과적으로 대처할 수 있다.

본 논문에서 제안한 방법은 상당히 많은 계산양을 요구한다. 그런데 페이지 교환 작업은 매우 빈번히 일어나는 작업이므로 현재의 하드웨어 환경에서는 성능의 향상을 기대하기 어렵다. 그러나 장차 퍼지 연산을 위한 하드웨어가 보조처리 장치로 지원된다면 시스템의 성능향상을 기대할 수 있을 것이다. 또한 본 논문에서 기술한 방법론이 여러가지 발견적인(heuristic)조건을 반영해야할 필요가 있는 스케쥴링 분야에 적용될 수 있으리라고 본다.

5. 참고문헌

[1] L.A.Zadeh, "The Role of Fuzzy Logic in the Management of Uncertainty in Expert System", *Fuzzy sets and Systems 11*, 1983, pp 199-227

[2] W.Pedrycz, *Fuzzy Control and Fuzzy Systems*, Research Studies Press Ltd., 1989, pp 61-80

Copyright (C) 2005 NuriMedia Co., Ltd.

[3] S Sahni *Concepts in Discrete Mathematics*, 2nd edition, The Camaot Publishing Company, 1985, pp163-191

[4] D Dubois & H Prade, *Fuzzy sets and systems*, Academic Press, 1980

[5] G J Klir & T A Folger, *Fuzzy sets, Uncertainty and Information*, Prentice-Hall International, 1988

[6] L A Zadeh, "Fuzzy sets", *Information and Control*, Vol 8, Academic Press 1965, pp 338-353

[7] J L Peterson & A Silberschatz, *Operating System Concepts*, 2nd edition, Addison Wesley, 1985, pp 201-255

Copyright (C) 2005 NuriMedia Co., Ltd.