재결정 위상의 분산적 구성과 비구조적 피어투피어 망에서의 효율적 검색

(Distributed Construction of the Recrystallization Topology and Efficient Searching in the Unstructured Peer-to-Peer Network)

박 재 현 †

(Jae-Hyun Park)

요 약 본 논문에서 비구조적 피어투피어 망을 위한 적은 검색 시간을 가지는 최적화된 위상을 구성하는 분산된 위상 제어 알고리즘을 제안한다. 각 노드는 높은 검색 적중률을 가지는 최적의 노드들을 노드 자신의 적중률에 지수적으로 비례하는 수만큼 선택하고, 그들과 연계한다. 총체적 거동은 자연계에서는 볼 수 있는, 각 입자의 에너지 준위에 따라 입자들이 결합되는 재결정 현상과 결과적으로 거의 유사하다. 구성된 위상의 노드들의 적중율들 사이에는 부분 순서(Partial-order) 관계가 있다. 그러므로, 질의 메시지가 노드를 방문하는 경우에, 그 노드는 항상 직전에 방문하였던 노드들 보다 더 높은 적중률을 가지고 있다. 또한, 무위도식(Freeloader) 노드로부터 보내진 질의 메시지는 한 홉 전달에 의해, 무위도식하지 않은 노드들로 전달될 수 있고, 그것은 다시는 무위도식하는 노드들을 방문하지 않는다. 이처럼 검색은 제한된 지연시간 안에 이루어진다. 또한, 본 논문에서는 이 위상을 활용하여 효과적인 연쇄반응적 검색 방법을 제안한다. 그러한 제어된 다중 전송 방식은, 방송을 사용하는 방식 보다 질의 메시지들의 수를 43 퍼센트만큼 줄이며, 검색시간을 94 퍼센트 절감한다. 제안된 방안의 검색 성공률은 99 퍼센트 이다.

키워드 : 위상 제어 프로토콜, 피어투피어 망, 재결정 위상, 검색

Abstract In this paper, we present a distributed topology control algorithm for constructing an optimized topology having a minimal search-time in unstructured peer-to-peer network. According to the proposed algorithm, each node selects the best nodes having higher hit-ratio than other nodes as many as the number being exponentially proportional to the hit-ratio of the node itself, and then it connects to them. The ensemble behavior of the proposed algorithm is very similar to the recrystrallizing phenomenon that is observed in nature. There is a partial order relationship among the hit-ratios of most nodes of constructed topology. Therefore once query message visits a node, it has a higher hit-ratio than the node that was visited last by the message. The query message even sent from freeloader can escape to the node having high hit-ratio by one hop forwarding, and it never revisits any freeloader again. Thus the search can be completed within a limited search time. We also propose the Chain-reactive search scheme using the constructed topology. Such a controlled multicasting reduces the query messages by 43 percent compared to that of the naïve Gnutella using broadcasting, while it saves the search time by 94 percent. The search success rate of the proposed scheme is 99 percent.

Key words : Topology Control Protocol, Peer-to-peer Network, Recrystallization Topology, Search

† 논문은 2007년도 중앙대학교 학술연구비(일반연구비) 지원에 의한 것임

† 정 회 원 : 중앙대학교 컴퓨터공학부 교수
hyunie@cau.ac.kr

논문접수 : 2007년 11월 1일
심사완료 : 2008년 4월 20일

정보과학회논문지: 정보통신 제35권 제4호(2008.8)

1. 서 론

많은 자연계의 망들, 예를 들어 하나의 세포 안에 있는 분자들, 생태계의 종들, 사회적 무리 안에서의 사람들과 같은 망들은 L 연결들을 가진 노드들의 수가 L^{-k}에 비례하는 척도-없는(Scale-free) 거듭제곱법칙(Power-law)의 망들이다(여기서 k는 망에 의존적인 상수이다)[1]. 통신 망들, 특히 비구조적 피어투피어(Peer-to-peer) 망은 또한 척도-없는 거듭제곱법칙 망이며, 이러한 위상에서 소수의 노드들이 많은 연결들을 가지며, 통신과 망 구성에 있어 중심축의 중요한 역할을 한다[2]. 거듭제곱법칙 망들의 이러한 특성들을 활용하여, 높은 차수의 노드들을 활용하는 효율적인 지역적 검색 전략들 제안되어 왔다[2].

그런데, 시스템 설계자는 이 생성된 위상보다 더 효과적인 망 위상을 설계할 수 있으며, 효과적인 검색 메커니즘을 만들기 위해 그것을 활용할 수 있다[3]. 최적화된 위상을 구성하거나, 질의 경로를 찾기 위해, 각 피어(Peer)가 적절한 피어를 선택하는 메커니즘을 설계하는 것은 효과적인 피어투피어 검색 시스템을 구성하기 위해 중요하다[4]. 구조적 피어투피어 망은 유일한 자원 각각을 망 안의 특정 노드에 사상한다. 이는 검색을 효율적으로 할 수 있도록 하지만, 유연성 부족을 가져오며, 이는 망의 신뢰성 문제와 데이타 관리의 자율성 이슈들을 야기한다[3,4].

본 논문에서는, 비구조적 피어투피어 망에서 최소 탐색시간을 가지는 최적화된 위상, 다시 말해 차수 서열(Degree sequence)이 거듭제곱 분포에 따르는 척도-없는 위상을 구성하는 병렬 분산적 알고리즘을 제안한다. 그리고, 구성된 위상의 성질을 활용하는 효율적인 검색 메커니즘을 제안한다. 제안된 방법은 특정 자원을 제공하는 피어를 발견하는 데 필요한 시간, 즉 검색 시간을 최소화 하고, 검색 과정에 의해서 발생하는 메시지들의 수, 즉 검색 비용을 줄인다.

볼츠만 기계는 위상의 구성과 같은 조합적 최적화 문제들을 풀기 위해서 널리 사용되어 왔다. 본 연구에서는 피어투피어 망의 노드들이 처리능력과 방송능력을 가지기 때문에 볼츠만 기계의 노드로서 적합하다는 사실에 주목하였다. 즉, 피어투피어 망의 노드들을 병렬 볼츠만 기계의 노드들로 사상함으로써, 우리들은 완전 분산된 위상 제어 알고리즘을 개발하였다. 제안된 알고리즘에 따라, 노드들은 가장 뛰어난 피어들을 선택하고, 검색 시간의 관점에서 전역적으로 최적화된 안정된 위상을 만든다. 그리고, 효율적인 탐색을 위해 이를 사용한다.

재결정 현상은 풀림이 일어나는 동안 금속성 물질에서 관찰된다[5]. 제안된 위상 제어 방법의 동역학은 각 분자가 잠재하는 에너지 준위에 따라 다른 분자들과 결합되는 재결정 현상과 결과적으로 매우 유사하다. 각 피어들은 알고 있는 한 다른 노드들 보다 더욱 높은 적중률을 가지고 있는 최고의 노드들과 연합한다. 유사하게 피어들과의 연합들의 수는 자신의 잠재 에너지 준위, 즉 자신의 적중율에 지수적으로 비례한다. 피어들의 적중율은 지수적으로 분포되고[6], 따라서 이는 볼츠만 요소이다. 그러므로 피어의 총체적 회합의 동적 움직임은 안정된 위상을 만든다. 피어의 잠재적 에너지의 척도로 적중률을 사용하였으나, 다른 척도들, 예를 들어 제공할 수 있는 대역폭도 사용할 수 있다.

구성된 위상의 노드들 사이에는 노드들의 잠재 에너지들에 대해서 부분 순서 관계(Partial Order Relationship)가 존재한다. 따라서, 극소치를 만날 때까지 척도에 대한 기울기(Gradient)를 최대로 하는 부분 최소치를 반복하여 선택하는 언덕-오르기(Hill-climbing)과 같은 간단한 탐색방법으로도 최적 혹은 부최적(Sub-optimal) 노드를 찾을 수 있다. 또한 그 위상에 격자와 같이 부분 순서가 있으므로, 제한된 횟수의 탐색 만으로 탐색에 성공할 수 있다.

재결정 위상의 노드들 사이의 부분 순서를 활용함으로써, 우리들은 제어된 다중 전송 방식(Controlled Multi-casting)을 사용하는 효율적인 검색방법을 제안한다. 이는 검색 비용 측면에서 좋은 랜덤 워크 검색의 장점과 검색 시간 측면에서 좋은 플러딩 검색을 효율적으로 결합한 검색 방법으로서, Kleinlock의 최적 검색 성능에 대한 분석 연구[3]의 결과를 일반화한 것이다. 제안된 방법은 검색시간과 검색비용 측면에서 효율적으로 검색을 수행한다.

2. 관련된 연구

오버레이(Overlay) 망의 위상의 특성은 피어투피어 망에서의 검색의 효율에 매우 중요한 영향을 미친다[7]. 확장성있고, 효율적인, 그리고 강인한 오버레이 위상들을 형성하기 위하여, 적응적 피어투피어 위상 프로토콜(Adaptive Peer-to-Peer topologies Protocol)이 제안되었다[8]. 피어들은 지난 거래들에 의해서 정의되는 자신의 지역적 신뢰도에 기초하여 그리디(Greedy)한 방법으로 연결들을 형성함으로써 망의 진정한 가치(trust-worthiness)를 개선한다[8].

그러나, ATP는 국지적 신뢰도(local trust)와 연결 신뢰도(connection trust)의 차이를 고려하지 않았다[9]. 한번 어떤 피어가 실수로 악의있는 피어의 가짜 파일들의 배포 통로로 사용되면, 그 피어는 그것의 이웃 피어들로부터 연결이 단절되며, 그 피어는 그것의 이웃들에게 많은 진품 파일들을 제공하였더라도 이 이웃과는 연

결되지 못할 것이다[9].

이를 해결하기 위해서, 상호 능력 기반 적응적 위상 프로토콜(Reciprocal Capacity based Adaptive Topology Protocol)이 제안되었다. RC-ATP로 동작하는 피어들은 피어의 직접 및 간접적으로 제공하는 서비스의 능력에 기반하여 이웃 피어들을 선택한다[9]. 그러나, 이 프로토콜 역시 피어는 그 자신이 평가한 신뢰도에만 의존하여 이웃 피어들을 선택함으로써, 제시된 결과에 의하면 검색 성공률이 최대 약 70%인 한계를 가진다.

그런데, 최근에 실재하는 망들에서의 최대 검색 성능은 위상이 척도-없는(거듭제곱 법칙) 망인 경우 달성될 수 있고, 이 경우 제공되는 검색 성능은 $O(\ln\ln N)$임이 밝혀졌다[10,11]. Merugu의 연구[12]에서, 피어투피어 망에서 작은세계(small-world) 위상을 구성하는 방법이 연구되었다. 이 연구에서 그러한 오버레이 위상을 구성하는 간단한 그리디 알고리즘을 제안하였는데, 이 알고리즘에 따르는 각 노드는 그들의 이웃들을 선택하기 위해서 독립적이고 분산적인 방법으로 동작한다. 그렇지만, 구성된 위상에서의 검색 성능은 모든 노드들의 이웃들이 임의로 선택되는 임의 그래프(random graph)의 그것보다 낮다. P. Dasgupta[11]는 작은세계 (small-world) 망을 유지하는 다중 에이전트(multi-agent) 기반의 피어투피어 망 구성 프로토콜을 제안하였다. 그러나 구성된 위상에서의 검색 성공율은 단지 20-60%이다.

최근에, H. Guclu[7]는 새로운 척도-없는 위상 생성 메커니즘을 제시하였다. 이 방법은, 전역적인 위상 정보를 사용하는 전통적인 척도-없는 위상 생성방법들과는 다르게, 완전하거나 또는 부분적으로 지역적인 정보를 사용한다. 그러나 더 나은 검색성능을 가진 위상들을 구성하기 위해서는, 이 메커니즘은 메시지가 폐기되기 전에 겪는 전달 횟수의 한계인 TTL을 크게 함으로써, 전역적 위상에 관한 많은 정보를 수집하는 것을 사실상 필요로 한다. 다시 말해, 제안된 메커니즘은 많은 메시지들의 전송을 필요로 한다. 그러나, 이렇게 위상을 구성함에 있어 전체 위상에 관한 완전한 혹은 부분적인 전역 정보를 필요로 하는 것은 오버레이 위상들을 구성하기 위해서 이러한 메커니즘을 사용하는 것을 매우 어렵게 한다.

사회적 망들에 관한 연구들은 관련된 정보의 발견이라는 과제는 적절한 사람들에게 문의함으로써 줄일 수 있다고 암시한다. 적절한 사람들은 원하는 정보를 가지고 있어서 관련된 내용을 직접 제공할 수 있거나, 적절한 사람들을 추천할 수 있는 사람이다. 이러한 망에서, 질의는 목적지에 가까워지도록 하는 적절한 사람들로의 연결들을 따라 전달된다[4]. INGA는 이러한 사회적 망들에서의 질의 경로 배정에 관한 관찰에 착안하여 만들어진 질의 메시지의 경로 배정(Routing) 알고리즘이다[4]. 이 시스템의 질의 경로 배정은 이웃들의 전문가적인 견해에 관한 지역적으로 가용한 지식과 의미론에 바탕을 둔 피어 선택 기능에 의해서 만들어진다.

본 논문에서 피어들 간에 다면평가(multisource feedback, 다시 말해 360 degree feedback)를 사용하여 척도-없는 망을 구성하는 완전히 분산된 위상 제어 프로토콜을 제시한다. 구성된 위상은 노드들의 검색 능력 간에 부분 순서가 존재하는 척도-없는 망이다. 각 노드의 연결들의 수는 자신의 검색 능력에 지수적으로 비례한다. 따라서 구성된 망의 노드들의 차수 서열(Degree sequence)의 분포는 거듭제곱 법칙에 따르며, 결과적으로, Cohen의 연구[10]에서 제시된 척도-없는 망의 구성 과정 모형으로 설명되듯이, 각 피어는 자신과 유사한 검색 능력을 가지는 다른 피어들과 연결하여 연합한다. 본 논문에서 구성된 위상이 검색 능력에 전역적으로 최적인 망을 구성하는 사실을 설명하기 위해서, 제시된 프로토콜과 볼츠만 기계와의 연결을 밝히고, 위상 최적화 문제를 볼츠만 기계 관점에서 모형화 했다.

제안한 프로토콜은, 다면평가 결과를 전파하는 메시지를 보내기 위해서, 자신의 능력에 지수적으로 비례하는 TTL을 사용하는 제어된 다중 전송방식을 사용함으로써, 효과적으로 다면평가의 결과를 전파하면서도, 상대적으로 적은 메시지들을 가지고 효율적으로 척도-없는 망의 위상을 구성한다. 또한, 구성된 위상의 특성들을 활용하는 효율적인 검색 프로토콜을 제안한다. 검색 성공률은 99 퍼센트 이다. 제시된 제어된 다중 전송 방식의 검색은 방송을 사용하는 방식보다 질의 메시지 수를 43 퍼센트만큼 줄인다. 제안된 검색 프로토콜이 원천 노드들에서 만들어진 원 질의 메시지의 11 퍼센트에 불과한 추가 메시지들을 만든다는 사실에 유의할 필요가 있다.

그밖에 오버레이 망을 구성하는 연구들로는 구조화된 피어투피어 시스템들이 있다. Chord는 집중형 색인을 사용하지 않고, 특정 피어에게 특정한 색인 정보를 배포하고, 검색 메시지를 그 피어로 라우팅하는, 분산 해시표 이다. Contents Addressable Network와 Tapestry는 의미론적인 질의 경로 배정을 사용한 구조적 피어투피어 시스템으로써 제안되었다. DHT 시스템은 많이 중복된 파일들이 요청되었을 경우에 비효율적이다. 실제 환경에서 방송(Flooding)에 의한 방법보다 나쁜 성능을 보이고, 망에 변화(Churn)가 심한 경우에는 더욱 나빠진다[4].

3. 재결정 위상을 구성하는 분산된 병렬 알고리즘과 검색

볼츠만 기계[13,14]는 최적화 문제에 대해 전역해를

구하기 위해서 널리 사용되어 왔다[14-17]. 그런데, 피어투피어 망의 노드들은 처리 능력과 방송 능력을 가지고 있어서, 볼츠만 기계의 뉴런의 역할을 수행하는데 아주 적합하다. 따라서, 목적 함수를 적절하게 유도한다면, 우리들은 이들 노드들을 가지고 병렬 볼츠만 기계를 구성할 수 있다.

3.1 공식화와 볼츠만 기계

먼저 우리들은 볼츠만 기계를 사용하여 위상 구성 문제를 공식화 한다. 이러한 문제의 공식화는 Cordie의 연구[8]에 있는 문제의 공식화와 상당히 유사하다. 그러나, 본 논문에서 망의 차수 서열의 분포가 볼츠만 분포, 다르게 모형화하면 거듭제곱 법칙에 따라야 한다는 사실을 추가하여 이러한 문제 정의를 볼츠만 기계를 사용하여 확장하였다. 이를 사용하여, 비구조적 피어투피어 망의 위상을 전역적으로 최적화하는 프로토콜을 제공한다.

볼츠만 기계는 가중치를 준 입력들의 합에 의거하여 확률적(Stochastically)으로 활성화하는 뉴런들로 구성된, 대칭적 회귀(Recurrent) 연결들을 가지는 신경망이다[13]. 구체적으로, n 개의 뉴런들로 구성된 신경망을 생각해보자. 가중치 w_{ij} 와 가중치 w_{ji}가 뉴런 i와 뉴런 j 사이의 대칭적 연결의 가중치이고, $w_{ij} = w_{ji}$ 라고 가정하자. 그리고, 회귀 연결을 가지는 뉴런의 가중치 w_{ii}는 0이라고 가정하자. 각 뉴런은 다음 2가지 값들 사이의 연속적인 구간에 있는 실수 값을 갖는다: 1 (활성) 그리고 0 (비활성). 그러면 i 번째 뉴런의 퍼텐셜(Potential), ε_i, 은 다음으로 정의된다.

$$\varepsilon_i = \sum_j w_{ij} x_j \tag{1}$$

여기서 x_j는 j 번째 뉴런의 상태이다[13].

이제 피어투피어 망의 노드들과 연결들을 사용하여, 신경망의 뉴런들과 연결들을 바로 만들 수 있다. 가중치 w_{ij}를 피어투피어 망의 노드 i와 노드 j 사이의 연결을 표현하기 위해 사용하고, x_j를 노드 i에 연결된 노드 j의 적중률을 나타내기 위해서 사용하자. 그러면, 식 (1)에서 뉴런 i의 퍼텐셜, ε_i, 는 피어투피어 망에서의 검색 연산에 대한 노드 i의 잠재적 능력을 의미하게 된다.

각 뉴런은 비동기적으로 ε_i에 종속하여 상태를 변화시킨다[13]. 따라서, 뉴런들의 상태들로 구성되는 벡터 $X = \{x_1, ..., x_n\}$로 볼츠만 기계의 상태를 표현할 수 있다. 따라서, 상태 천이는 벡터 X의 2^n 상태들로 구성된 마코프 연결로서 수학적으로 기술된다. 모든 뉴런들이 연결되는 경우에, 이것은 하나의 안정적인(얼고딕) 마코프 연결을 구성하는데, 이 마코프 연결은 독특한 안정적 분포 $p(X)$를 가진다. 이러한 안정적 분포는 일반적인 경우에 다음과 같이 주어진다[13,18].

$$p(X) = c_1 \cdot e^{E(X)} \tag{2}$$

단, $p(X)$의 최대 값이 존재한다고 가정한다. 여기서 c_1 은 정규화 상수이다. 그리고

$$E(X) = \sum_i \varepsilon_i \tag{3}$$

는 망의 형상 X의 총 잠재에너지를 나타낸다. 식 (2)는 일반적인 볼츠만 요소의 듀얼(Dual)이다.

따라서, 볼츠만 기계가 무슨 상태에서 시작하든지, 그것이 상태 X에 있을 확률은 $p(X)$에 수렴한다. 다시 말해, 오랜 경과 시간 후의 형상 X의 상대 빈도수는 확률 $p(X)$로 표현될 수 있다[13,18]. 그러므로, 우리들은 다음의 보조정리를 얻을 수 있다.

보조정리 1. 볼츠만 기계의 전체 에너지가 평형상태로 전역에서 수렴하기 위한 충분 조건은, 형상 X의 노드들의 총 잠재에너지 $E(X)$ 분포가 $p(X) = c_1 \cdot e^{E(X)}$와 같은 볼츠만 분포인 것이다. 단, 이 경우 $p(X)$의 최대 값은 존재함을 가정한다. ■

많은 분자로 구성된 거시계가 열적 평형에 있을 때, 분자들은 미시적 에너지 상태에서 에너지 상태에 대한 볼츠만 인수로 분포된다[18]. 그러므로, 거시계가 열적 평형을 이루고 있는 경우에, 계의 분자가 상태 s에 있어 ε_s의 퍼텐셜을 가질 확률은 다음과 같이 표현된다[18].

$$p(\varepsilon_s) = c_2 \cdot e^{-\varepsilon_s} \tag{4}$$

여기서 c_1 은 정규화 상수이다.

그러므로, 우리들은 다음의 보조정리를 얻을 수 있다.

보조정리 2. 볼츠만 기계의 전체 에너지가 평형상태로 전역에서 수렴하기 위한 충분 조건은, 계의 노드가 상태 s에 있어 ε_s의 퍼텐셜을 가질 확률 $p(\varepsilon_s)$이, $p(\varepsilon_s)$의 최대 값이 존재한다는 가정 하에, 다음과 같은 볼츠만 분포를 가지는 것이다.

$$p(\varepsilon_s) = c_3 \cdot e^{\varepsilon_s} \tag{5}$$

여기서 c_3 은 정규화 상수 이다. ■

그러므로, 우리들은 다음의 정리를 얻을 수 있다.

정리 1. 비구조적 피어투피어 망의 노드들로 구성되는 병렬 볼츠만 기계의 전체 검색을 위한 에너지가 평형상태에서 전역적 최대로 수렴하기 위한 충분 조건은, 계의 노드가 상태 s에 있어 ε_s의 퍼텐셜을 가질 확률 $p(\varepsilon_s)$이, $p(\varepsilon_s)$의 최대 값이 존재한다는 가정 하에, 다음 $p(\varepsilon_s) = c_3 \cdot e^{\varepsilon_s}$ 와 같은 볼츠만 분포를 가지고, 전체 망의 검색을 위한 총 퍼텐셜 에너지 $\boldsymbol{E(X)}$가 최대화 되는 것이다.

증명. 보조정리 2에 의거하여, 증명되었다. ■

검색 퍼텐셜 에너지 $E(X)$를 최대화시키기 위해서는,

식 (3)에 의해서, 각 노드의 퍼텐셜, ε_i,가 최대가 되어야 한다. 각 노드의 퍼텐셜, ε_i,가 최대가 되기 위해서는, 식 (1)에 의해서, 각 노드의 검색에 유용한 연결들의 개수를 최대화 하여야 한다.

따름정리 1. 비구조적 피어투피어 망의 노드들로 구성되는 병렬 볼츠만 기계의 전체 검색을 위한 에너지가 평형상태에서 전역적 최대로 수렴하기 위한 충분 조건은, 계의 노드가 상태 s에 있어 ε_s의 퍼텐셜을 가질 확률 $p(\varepsilon_s)$이, $p(\varepsilon_s)$의 최대 값이 존재한다는 가정 하에, 다음 $p(\varepsilon_s)=c_3\cdot e^{\varepsilon_s}$ 와 같은 볼츠만 분포를 가지고, 각 노드의 검색에 유용한 연결들의 개수를 최대화 하는 것이다. ■

그러므로, 높은 검색 성능을 가진 안정적 위상을 구성하는 문제는 다음과 같이 정리할 수 있다.

정리 2. 검색 능력이 전역적 최대에 수렴하는 안정적 위상 구성 문제는 다음과 같이 기술할 수 있다. 단, 이 경우 $p(X)$와 $p(\varepsilon_s)$의 최대 값은 존재함을 가정한다.

다음을 최대화한다:

$$E(X)=\sum_i \varepsilon_i$$

다음 조건을 전제로 한다:

$$p(X)=c_1\cdot e^{E(X)} \vee p(\varepsilon_s)=c_3\cdot e^{\varepsilon_s}, \qquad (6)$$

여기서 ε_i는 다음과 같이 정의된다:

$$\varepsilon_i=\sum_j w_{ij}x_j$$

증명. 보조정리 1과 보조정리2, 그리고 정리 1에 의거하여, 증명되었다. ■

따라서, 이러한 위상 구성 문제를 다음과 같이 풀 수 있다. (1) 볼츠만 기계의 각각의 노드들을 비구조적 피어투피어 망의 각 노드들에 사상한다. (2) 각 노드의 연결 제어 논리를 따름정리 1과 정리 2에 따라 만든다.

3.2 재결정 위상의 구성을 위한 볼츠만 기계

이제, 부분 순서를 가지는 효과적인 위상의 피어투피어 망을 구성하기 위한 목적 함수를 유도한다. 피어투피어망의 노드들의 적중률들의 분포는 식 (2)와 같은 볼츠만 분포이다[6,18]. 그러므로, 보조정리 2에 따라, 노드들 간의 연결여부를 결정하는 각노드의 연결제어 논리에서 적중률을 사용하여 전역적으로 수렴하는 안정적인 위상을 만들 수 있다. 더욱이, 정리 2에 따라서 확률 $p(\varepsilon_s)$가 식 (5)와 같은 볼츠만 분포를 가지도록 만들면, 다시 말해 적중률을 입력 변수로 하여 식 (7)과 같은 볼츠만 분포를 만들면, 노드들의 적중률의 분포가 장래에 다른 분포로 변하더라도, 정리 2에 의거하여, 전역적으로 수렴하는 안정적인 위상을 만들 수 있다.

적중률을 사용하여, 노드가 검색에 유용한 정도를 측정할 수 있다. 따름정리 1 에 따라서, 각 노드가 검색에 유용한 연결들의 수를 최대화 하기 위해서, 자신이 알고 있는 노드들 중에서 적중률이 높은 순서로 연결을 가지도록 각 피어의 연결 제어 논리를 구성함으로써, 결과적으로 부분 순서를 가진 재결정 위상을 구성할 수 있다. 그러므로, 다음과 같은 정리가 성립한다.

정리 3. 비구조적 피어투피어 망이 효율적인 검색에 효과적인 부분 순서를 가진 재결정 위상의 평형상태로 수렴하기 위한 충분조건은 확률 $p(\varepsilon_s)$가 볼츠만 분포를 가지고 각 노드가 검색에 유용한 연결들의 수를 최대화 하는 것이다.

증명. 보조정리 1과 보조정리 2, 그리고 따름정리 1과 정리 2에 의거하여, 증명되었다. ■

이제 볼츠만 기계를 사용하여 검색 능력이 전역적 최대에 수렴하는 안정적인 위상을 만들 수 있다. 먼저, 검색에 효과적인, 검색에서 성공하는 비율, 다시말해 적중률이 높은 노드들의 퍼지 집합을 정의한다. 그리고, 다음과 같이 노드가 얼마나 검색에 기여할 수 있는지를 표현하는 볼츠만 요소 로서 회원 여부를 결정하는 소속함수 $p[\varepsilon_i = hit_ratio]$를 정의한다. 이 함수는 검색 기여도가 최대가 되는 경우에 극대점을 갖는다.

$$p[\varepsilon_i = hit\_ratio]=c_4\cdot e^{w\cdot hit\_ratio} \qquad (7)$$

이고, ε_i는 노드 i의 검색 연산을 위한 기여 퍼텐셜이다. w는 정규화 상수이고, w의 값으로 2.28을 사용한다[19]. C_4는 설계 여유를 표현하는 상수로써, 본 연구에서는 설계 여유를 두지않고, 값으로 1을 사용한다. 식 (7)에 의해서 검색 기여도가 높은 노드들이 많은 연결들을 가지도록 된다.

정리 4. 비구조적 피어투피어 망의 효율적인 검색에 효과적인 부분 순서를 가진 재결정 위상을 다음에 의해서 구성될 수 있다. (1) 각 노드가 볼츠만 분포를 가지는 $p[\varepsilon_i = hit_ratio]=e^{w\cdot hit_ratio}$에 비례하여 연결들을 가지도록 각 피어의 연결 제어 논리를 만든다. 또한, (2) 각 노드의 검색에 유용한 연결들의 수를 최대로 만들기 위하여, 노드 자신이 알고 있는 노드들 중에서 적중률이 높은 순서로 연결을 만들도록 연결 제어 논리를 구성한다.

증명. 식 (7)과 따름정리 1, 그리고 정리 3에 의해서 증명되었다.■

3.3 프로토콜들

본 절에서 안한 프로토콜들의 상세 설계를 제시한다. 이를 위해서 가용한 피어들에게 연결하는 부트스트랩 프로토콜을 먼저 살펴보고, 재결정 위상의 구성 프로토콜을 제시하고, 질의 검색 프로토콜을 설명한다.

```
X.bootstrap() // Bootstrapping
Require: Set knownServents = {}, float hit_ratio = 0.75, int maxDegree = c4 · e^(w·hit_ratio),
Set ConnectedPeersList, Set BlackPeersList, Set OpenSocketsList
Ensure: ConnectedPeersList.size < maxDegree
1: knownServents ← BootServer.reg_request_ServentsList(X)
2: for each Si ∈ knownServents do
3:      if Si ∈ BlackPeersList then continue
4:      end if
5:      if Si ∈ ConnectedPeersList then
6:              update HitRatio of Si in ConnectedPeersList, OpenSocketsList,
                        with Si.HitRatio
7:      else
8:              X connects to Si
9:              if Si accepts X then
10:                     add Si to ConnectedPeersList, and OpenSocketsList
11:             end if
12:     end if
13:     X. pruning_nodes_having_lesser_potential_energy()
14:     if X is not bootserver and ConnectedPeersList.size < maxDegree then
15:             X.bootstrap() // get more servent names
16:     end if
17: end for

X.pruning_nodes_having_lesser_potential_energy()
Require: int maxDegree = c4 · e^(w·hit_ratio), Set ConnectedPeersList, List OpenSocketsList,
        // Pruning the connection for neighbors who have lesser potentials for search
1: while OpenSocketsList > maxDegree do
2:      X disconnects OpenSocketsList.lesserHitRatioPeer
3:      remove lesserHitRatioPeer from OpenSocketsList and ConnectedPeersList
4: end while

BootServer.reg_request_ServentsList(X) //Registering X and requesting ServentLists
Require: FIFO knownServentsFIFOList= {}, Set AvailableServentsList, Int numberOfAllocatedServents
=0, int MaxNumberOfRequestedServents = k
1: knownServertsFIFOList ← knownServertsFIFOList + X
2: while (numberOfAllocatedServents < MaxNumberOfRequestedServents) do
2:      if Yi.ThereIsAvaiableSocket() s.t. ∀Yi ∈knownServentsFIFOList  then
3:              AvailableServentsList ←  AvailableServentsList ∪ {Yi}
4:              numberOfAllocatedServents ++
5:      end if
6: end while
7: return AvailableServentsList
```

그림 1 부트스트랩 프로토콜

3.3.1 부트스트랩 프로토콜

모든 노드들은 부팅 직후에는 검색 서비스가 가능한 피어의 리스트를 가지고 있지 않으며, 자신의 검색 기여도를 알 수 없다. 노드들은 잘 알려진 부트스트랩 서버로부터, 피어들의 리스트를 획득한다. 부트스트랩 서버들은 정책에 의해서 클러스터링을 할 수 있다. 다시 말해, 피어들의 리스트의 배포 순서에 따라 로컬 클러스터가 형성되며, 처음에는 일반적으로 이들 클러스터들이 인접한 클러스터들과 클러스터를 구성하는 링-형태의 위상을 구성하게 된다. 링 형태의 위상은 긴 지름 (Diameter)를 가지게 된다.

제안된 부트 스트랩 기능에 따라, 각 노드는 부트 서버로부터 획득한 피어들의 주소들 로의 접속을 시도한다. 재부팅하는 경우에는, 이전에 연결 요청을 거부한 피어들의 목록인 *BlackPeersList*에 등록되어 있는 주소들을 제외한 주소들로의 접속을 시도한다. 식 (7)에서

```
X.send_Ping_to_neighbors() //send ping mesg to neighbors periodically
Require:  Int  numberOfsentSocket, Set OpenSocketsList, Set knownServentsList, int TTL = c5 · e^(-w-hit_ratio)
1:        OpenSocketsList.send(pingMesg, TTL)

X.received_Ping_from_neighbors(Y, Mesg) //received ping mesg from neighbors
Require: Set knownServentsList,  int maxDegree = c4 · e^(w-hit_ratio)
1: if Mesg.ID ∈ seenDesclDList ∨ Y = X then
2:      return
3: else
4:      X.adaptive_forward_of_ping()
5:      TTL ← c4 · e^(w-hit_ratio)  // set  the TTL of Pong mesg as c4 · e^(w-hit_ratio)
6:      Y.sendPongs(knownServertsList , TTL) // send knownServentList to peer Y
7: end if

X.adaptive_forward_of_ping() //Adaptive ping while reducing the number of Ping messages
Require: int maxDegree = c4 · e^(w-hit_ratio)
Int  numberOfForwardedSocket = 0, List OpenSocketsList
// Choose the next hops for Adatptive walks
1: while (numberOfForwardedSockets ≤ max_degree) do
2:      if Z.HitRatio = max{Yi.HitRatio, ∀Yi.SocketID ∈OpenSocketsList} then
3:              Z.forward (pingMesg)
4:              numberOfForwardedSockets++
5:      end if
6: end while

X.received_Pong_from_neighbors(ServentsListFromPongMesg) //received pong mesg from neighbors
Require: Set knownServentsList, int maxDegree = c4 · e^(w-hit_ratio) , Set ConnectedPeersList, Set BlackPeersList, List OpenSocketsList,
Ensure: ConnectedPeersList.size < maxDegree
1: X.backroute(PongMesg) // backroute Pong message to the peer who initiated Ping message
2: for each Yi ∈ ServentsListFromPongMesg do
3:      if Yi ∈ BlackPeersList ∨ Yi ∈ ConnectedPeersList then
4:          continue
5:      else
6:          X connects to Yi
7:          if X connected to Yi
8:           then add Yi to ConnectedPeersList and OpenSocketsList
9:           else add Yi to BlackPeersList
10:         X. pruning_nodes_having_lesser_potential_energy()
11:     end if
12:end for

Y.connected_to(X) //Connected as best neighbors
Require: Set knownServentsList, int maxDegree = c4 · e^(w-hit_ratio) , Set ConnectedPeersList, Set BlackPeersList, Set OpenSocketsList,
Ensure: ConnectedPeersList.size < maxDegree
1: if X ∈ ConnectedPeersList then
2:      return
3: else
4:      Y accepts the connection from X
5:      add X to ConnectedPeersList and knownServentsList, and OpenSocketsList
6:      Y. pruning_nodes_having_lesser_potential_energy()
9: end if
```

그림 2 재결정 위상 구성 프로토콜

제시된 식에 hit_ratio 값으로 0.75를 적용하여 계산한 *maxDegree* 수 만큼 접속을 시도하며, 이를 초과하는 접속들은 *X.pruning_nodes_having_lesser_potential_energy*() 함수에 의해서 검색능력에 있어 상위의 노드들을 *maxDegree* 수 만큼 남겨두고 나머지 노드들과의 연결을 제거한다.

3.3.2 재결정 위상의 구성 프로토콜

이제 다면 평가(Multisource feedback)을 사용하는 척도-없는 위상의 재구성 프로토콜을 설명한다. 여러 피어들로부터 전달된 능력순에 의거하여 정렬한 피어들 목록들을 사용하는 다면 평가를 위해서, 각 노드가 유지하고 있는 '(피어주소, 평가된 능력)' 쌍으로 구성된 능력순으로 정렬된 피어들의 목록을 넣어 보낼 수 있도록 수정하였다.

즉, 피어주소 목록만으로 구성되었던 퐁 메시지 형식에 각 피어주소에 대응하는 '평가된 능력' 항을 추가하여 그 메시지 형식을 수정하였다.

제안된 프로토콜의 *received_Pong_from_neighbors*() 함수에 기술된 바와 같이, 도착하는 퐁 메시지에서 다른 노드의 피어들에 대한 평가 목록을 받아서, 연결을 거절하거나 연결에 실패한 피어의 리스트인 *BlackPeersList* 에 들거나, 이미 연결한 피어들의 리스트인 *Connected-PeersList*에 존재하지 않으면, 연결을 시도한다. 연결에 성공하면 자신의 피어들을 능력순으로 정렬한 목록인 *knownServentsList*를 연결한 피어의(피어주소, 능력치) 값을 사용하여 갱신한다. 다시말해, 이러한 다면 평가를 사용하여, 결과인 능력순으로 정렬한 자신의 피어 목록을 갱신한다. 연결에 실패하면, *BlackPeersList*에 넣고, 이는 들어간 피어가 시간이 지나면서 삭제되도록 *LRU* 알고리즘으로 관리한다.

그리고, 3.3.3절에 기술된 탐색 프로토콜에서와 같이, 이 목록은 자신이 요청한 트랜잭션이 탐색에 성공하여 돌아오는 경우에, 이를 전달한 직접연결된 이웃 피어들을 스스로 평가하여, 자신이 다중평가와 스스로의 평가를 통해 알고 있는 피어의 리스트인 *knownServents-List*와 연결된 이웃의 리스트인 *ConnectedPeersList*의 해당 피어 항목의 능력평가치를 각각 갱신한다. 그 다음, 갱신 결과를 사용하여 *max_degree* 개의 상대적으로 우수한 이웃들만을 유지하고, 나머지 우수하지 못한 이웃들에 대한 연결을 제거한다. *connected_to*() 함수에 기술된 바와 같이, 다면 평가 결과 우수하다고 판정되어, 연결을 요청 받은 피어는 연결을 받아준다. 그 다음, 연결을 받아준 피어도 역시 *max_degree* 갯수의 상대적으로 우수한 이웃만을 유지하고, 나머지 우수하지 못한 이웃들에 대한 연결을 제거한다.

다음에서 척도-없는 위상을 효과적으로 유지하면서, 위상제어 프로토콜의 메시지들의 수를 줄이는 방안을 설명한다. 각 노드는 주기적으로 *send_Ping_to_neighbors* () 함수를 사용하여 핑(Ping) 메시지를 발생 시킨다. 높은 적중률을 가진 노드는 가까운 이웃 노드들에게 다중 전송 방식으로 핑 메시지를 보내는 반면에, 낮은 적중률을 가진 노드는 *TTL*을 크게 설정해서 멀리 떨어진 노드에, 적중률을 가진 노드들보다 지수적으로 적은 수의 핑 메시지들을 보내게 된다. 다시말해, 각 노드는, 정리 4에 의거하여, *TTL*이 $c_5 \cdot e^{-w \cdot hit\_ratio}$ 로 설정된 핑 메시지들을 생성하여, 그것들을 식 (7) 즉 $c_4 \cdot e^{w \cdot hit\_ratio}$ 의 수만큼 다중방송을 사용하여 보낸다. 결과적으로, 낮은 적중률을 가진 노드는 멀리 떨어진 노드들에게 피어들의 목록을 요구하게 된다.

그림 3에 예시한 바와 같이, 이 핑 메시지는 각 중간 노드에 의해서 함수 *adaptive_forward_of_ping*()를 사용하여 전달된다. 낮은 적중률을 가진 노드 A가 높은 적중률을 가진 노드 B로 핑 메시지를 보내면, 단조 증가하는 적중률을 가진 중간노드들을 통해서 전달된다. 중간 노드들의 다중 전송되는 핑 메시지의 수는 정리 4와 일관성 있게 노드의 차수인 $c_4 \cdot e^{w \cdot hit\_ratio}$ 에 비례한다. 따라서, 그림 3과 같은 모양의 부분순서 그래프를 통해서, 핑 메시지는 전달된다. 퐁 메시지는, 핑 메시지가 전달된 경로의 반대 방향의 경로를 통하여, 노드 B 로부터 노드 A로 회신된다. 결과적으로, 그림 3과 같이 효율적인 부분순서 그래프 형태로 메시지를 전달하게 된다. 이 퐁 메시지가 전달되며, 노드 B와 노드 A 사이에 존재하는 중간노드들도 퐁 메시지에 첨부된 평가된 피어들 리스트의 피어들에 대해 연결들을 시도한다(이는 *received_Pong_from_neighbors*()에 잘 기술되어 있다).

한편 중간 노드들은 함수 *received_Ping_from_neighbors*()에 따라 자신의 평가를 포함한 다면평가 결

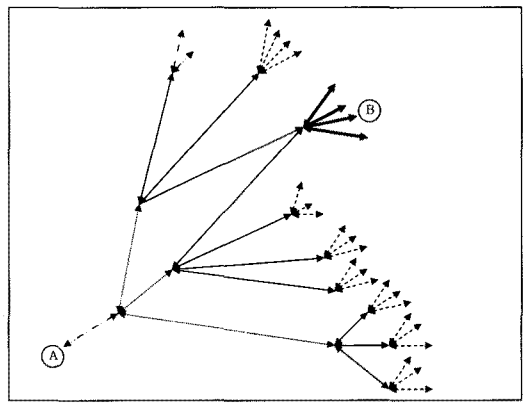

그림 3 재결정 위상 구성 프로토콜의 동작

과가 담겨있는 퐁 메시지를 만들어 이웃들에게 보낸다. 이 경우 TTL을 $c_4 \cdot e^{w\text{-}hit_ratio}$로 설정하여 보냄으로써, 높은 적중율을 가진 노드들로부터 보내진 퐁 메시지는 멀리 있는 노드들에 도달한다. 한편, 낮은 적중률을 가진 노드의 퐁 메시지는 적은 수의 연결들을 통해서 가까운 노드들에 한하여 전달된다. 다면평가 결과가 담긴 퐁 메시지를 받은 노드들은 이들 퐁 메시지들을 자신의 다면평가에 사용한다.

그림 3에 예시한 바와 같이, 적중률이 높은 노드 B는 커다란 TTL을 산출하여 퐁(Pong) 메시지에 설정함으로써, 그것이 핑 메시지를 보냈던 노드에 도달할 수 있도록 한다. 높은 적중률을 가진 노드는 자신의 다면평가 결과가 담긴 퐁 메시지를 원천지 노드까지 범위 안에 있는 많은 이웃 노드들에게 전달하여, 그들이 연결들을 시도하도록 한다. 다시 말해, 높은 적중률을 가진 노드들은 다면평가한 피어들의 목록을 멀리 배포하고, 핑 메시지를 전달한 중간 노드들도 그 목록을 알게 되어, 전달 경로에 중간에 있는 노드들은 이들 목록에 있는 우수한 피어들에게 연결을 시도한다. 낮은 적중률을 가진 노드들은 주변의 노드들의 국지적인 연결 요청만을 받게 된다.

우리들은 제안된 프로토콜의 이러한 거동을 재결정 현상의 재결정 알갱이의 형성과정을 가지고 은유적으로 설명할 수 있다. 재결정 현상은 풀림 동안 금속성 물질에서 관찰된다.

퍼텐셜이 높은 분자들이 멀리 에너지를 전달하며, 퍼텐셜이 높은 분자들과 재결정 핵을 형성한다 [5]. 유사하게, 적중률이 높은 노드들이 멀리 메시지를 보내며, 적중률이 높은 노드들과 결집하여 재결정 핵을 형성한다.

노드들의 연결 차수가 $c_4 \cdot e^{w\text{-}hit_ratio}$이므로, 구성된 재결정 위상은 제곱법칙의 망이다. 노드들은 자신들 보다 약간 낮은 적중률을 가진 노드들을 주로 연결한다. 그리고, 허용되는 수 이상의 적중률이 낮은 노드들로의 연결을 제거한다. 결과적으로, 제안된 프로토콜은 많은 퍼텐셜을 가지는 노드들로부터 적은 퍼텐셜을 가지는 노드들로 부분 순서를 표현하는 연결들을 노드들 간에 만든다.

제시된 프로토콜에 의해 구성된 위상에 3-코어연산(3-core operation)을 적용하면, 몇 개의 분할들을 발견할 수 있다. K-core는 모든 노드가 적어도 k 개의 노드들로 연결된 부분 그래프 이다. 1000개의 피어들로 구성된 망을 모의실험한 경우에 구성된 위상의 노드들의 대부분이 2개의 분할들을 구성한 것을 확인하였다. 그림 4에 보여진 것처럼, 분할의 노드들을 배치하기 위해서 Kamada-Kawai 알고리즘을 사용하면, 노드들은 동심원의 형태로 배치됨을 볼 수 있었다. 여기서, Kamada-Kawai 알고리즘은 널리 사용되는 힘에 의해 위치가 배정되는 알고리즘(force directed placement algorithm)이다. 이는 노드를 물체로 연결을 노드들을 연결하며 힘을 제공하는 스프링으로 보며, 지역 에너지가 최소화 될 때까지 노드들을 그들에 있는 힘에 따라 움직이며 그래

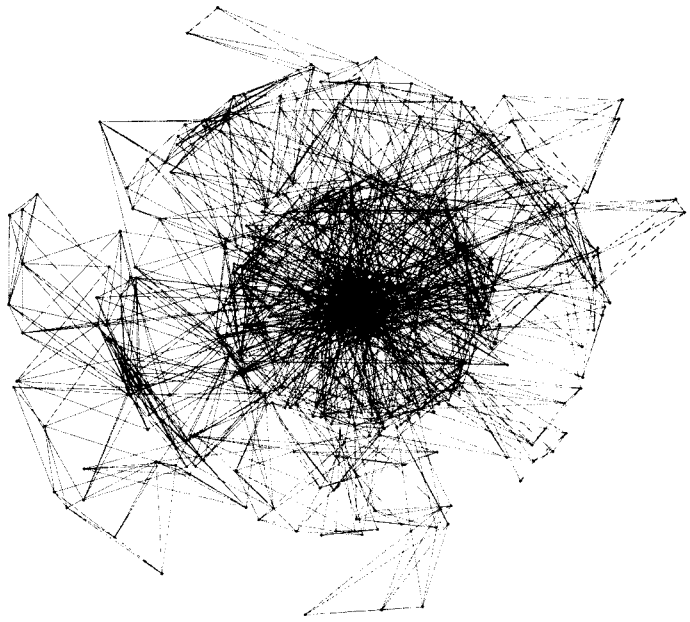

그림 4 Kamada-Kawai 알고리즘으로 배치한 재결정 위상의 3-core 분할

```
X. chain_reactive_query_sending() //send a query for Chain-reactive search
Require: Query q, int TTL = c5 · e^(-w·hit_ratio),
Set SelectedNextHops
// Choose the next hops for Chain-reactive walks
1: SelectedNextHops ← X.best_next_hops()
2: for each Si ∈ SelectedNextHops do
3:        Si.send(q, TTL)
4: end for

X.best_next_hops() //choose next hop nodes for Chain-reactive walks
Require: Set OpenSocketsList, Int numberOfSelectedNextHops=0, Set SelectedNextHops = {}, int
maxDegree = c4 · e^(w·hit_ratio)
1: while (numberOfSelectedNextHops < max_degree) do
// choose the neighbor which has maximum hit-ratio among neighbors that have not been selected as next
hop node.
2:   if Z.HitRatio = max{Yi.HitRatio, ∀ Yi ∈ OpenSocketsList ∧
                                Yi ∉ SelectedNextHops}
3:      then
4:           SelectedNextHops ← SelectedNextHops ∪ Z
5:           numberOfSelectedNextHops++
6:      end if
7: end while
8: return SelectedNextHops

X.find_target_or_chain_reactive_forwarding(Query q)
// find target file, or forwarding the query for Chain-reactive search
Require: Query q, int maxDegree = c4 · e^(w·hit_ratio), int TTL = c5 · e^(-w·hit_ratio), int hit_ratio,
Set SelectedNextHops
1: if X.where_target_found(q) then
      // Backroute the query hit message through the path through which the query has been passed.
2:    X.backroute(queryHitMesg)
3: else
       // Choose the next hops for Chain-reactive walks
4:    SelectedNextHops ← X.best_next_hops()
5:    for each Si ∈ SelectedNextHops do
6:          Si.forward(q)
7:    end for
8: end if
9: X.reevaluate(hit_ratio, X.where_target_found()) // Reevaluate my capability by myself

X. where_target_found(Query q) //check if X has the target document for q
1: if X.has relevant documents for q then return true
2: else return false end if

X.received_QueryHit_from_neighbors(QueryHitMesg qh, Set NeighborGivingQueryHitMesg)
//received QueryHit mesg from neighbors
Require: Set knownServentsList, int maxDegree = c4 · e^(w·hit_ratio), float estimatedHitRatio, Set
ConnectedPeersList, Set BlackPeersList, List OpenSocketsList,
Ensure: ConnectedPeersList.size < maxDegree
// Estimate the hit_ratio of neighbor by myself
1: estimatedHitRatio= NeighborGivingQueryHitMesg.estimate(HitRatio)
// Update hit_ratio of ConnectedPeerList and OpenSocketsList
2: update NeighborGivingQueryHitMesg.HitRatio in ConnectedPeersList, knownServentsList, and
                                                OpenSocketsList with estimatedHitRatio
3: X.use(qh) // Use QueryHit message
```

그림 5 제안한 질의 라우팅 프로토콜

프를 펼친다. 그림 4에 의해서, 이러한 분할이 적중률 퍼텐셜에 따라 부분 순서를 가짐을 알 수 있다. 이러한 척도-없는 망의 구성 과정은 Cohen의 연구[10]의 척도 없는 망의 구성 과정을 설명하는 모형과 일관성이 있다.

3.3.3 질의 검색 프로토콜

Kleinlock의 연구[3] 중 따름 정리 1은 낮은 복제본 밀도 클러스터들로부터의 검색을 위해 아주 큰 검색시간 비용을 감수한다면, 높은 복제본 밀도 클러스터에서 플러딩 검색을 사용하고 낮은 복제본 밀도 클러스터에서 임의적 보행자(Random walker) 검색을 사용하는 것이 좋음을 보여주고 있다. 본 논문에서는 만들어진 재결정 위상의 부분 순서 특성을 사용하여, 많은 검색시간의 비용 없이, 이 혼합된 전략을 사용하여 새로운 검색 알고리즘을 제시한다.

사용자가 검색을 노드에 요청하면, 그림 6에 예시하는 바와 같이, 그 노드 A는 *X.chain_reactive_query_sending*()를 사용함으로써, 이웃 노드들 중 가장 높은 적중률을 가진 노드들에게 $c_4 \cdot e^{w \cdot hit\_ratio}$ 만큼의 수의 질의 메시지들을 보낸다. 낮은 적중률을 가진 노드가 질의 메시지를 보내면, 이웃에서 목표 파일의 색인을 발견할 확률은 낮아서, 그림 6에 예시한 바와 같이, *TTL*을 크게함으로써 각노드는 멀리있는 노드들에게 질의 메시지를 보낸다. 한편, 높은 적중률을 가진 노드들이 이웃에서 목표 파일을 발견할 확률은 높다. 더군다나, 우리들은 다중전송 방식을 사용하여 질의 메시지들을 보냄으로써 목표 파일을 발견할 확률을 높일 수 있다.

find_target_or_chain_reactive_forwarding() 함수를 사용하는, 자신이 소유한 파일 인덱스에서 탐색이 성공하는 노드(예를 들어, 그림 6의 노드 B)는 질의 성공 메시지를 질의 메시지를 전달한 경로의 반대 방향의 경로를 통하여 질의를 발생시킨 피어에게(예를 들어, 그림 6의 노드 A) 보내고, 발견하지 못하는 경우 질의 메시지를 자신의 연결수 만큼 복제하여 전달한다. 그리고 나서, 자신의 자체 검색 능력을 재평가를 한다. 이러한 평가를 위해서는 [8]이나 [9]에서 사용된 방법들을 사용할 수 있다. 이러한 평가에 기초해 자신의 연결차수를 다시 결정한다.

received_QueryHit_from_neighbors() 함수는 탐색 성공 메시지를 받았을 경우에, 자신의 직접 연결된 이웃에 대한 평가를 다시 계산하고, (피어주소, 검색능력) 쌍들이 검색능력에 대해 정렬된 리스트인 *Connected-PeersList*와 동일한 쌍들의 다면평가의 결과를 담고있는 *knownServentsList*를 다시 계산된 값으로 갱신한다. 여기서, 자신의 능력을 스스로 평가한 것과 직접 연결된 이웃을 직접 평가한 것은 다면 평가의 기초자료 역할을 한다. 퐁 메시지에 담겨서 전달되는 이러한 기초 평가결

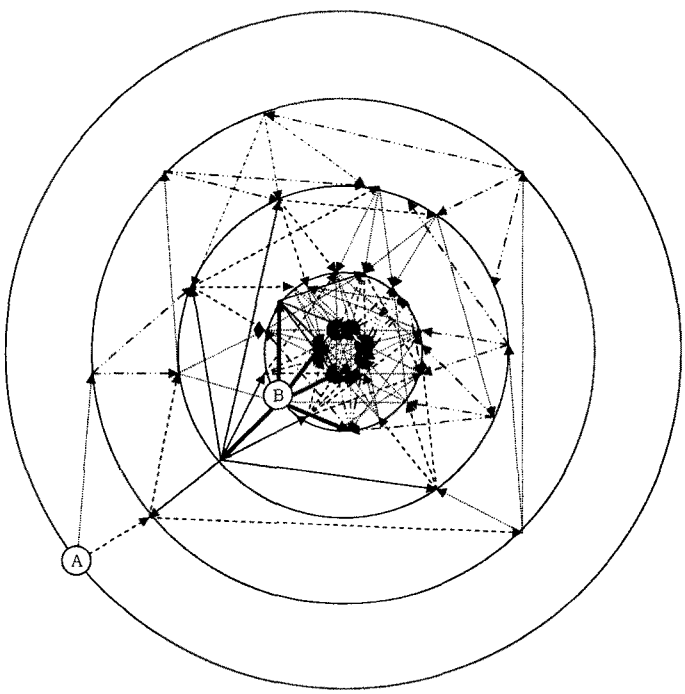

그림 6 재결정 위상 상에서의 연쇄 반응적 검색 프로토콜의 동작

과들을 다른 노드들이 다면 평가로 활용함에 유의하자.

또한 각 노드가 다수의 노드들에게 다중 메시지들을 보내는 대신에 최선의 노드에게 하나의 메시지를 보내도록 연쇄반응적 검색을 수정함으로써, 최선의 1-보행자(Best 1-walker) 라고 불리우는 검색법을 제시할 수 있다.

4. 성능분석

본 절에서 재결정 위상에서의 연쇄 반응적 검색의 성능을 설명하는 확률적 모형을 제시한다. 그리고, 이 분석적 결과들을 모의실험 결과들과 비교하여 검증한다.

4.1 확률적 모형

비구조적 피어투피어 망에서의 검색의 최적 성능을 분석하기 위해, 확률적 모형이 제시되었다 [3]. 비구조적 피어투피어 망에서의 주요 검색 대상인, 멀티미디어 파일들에 대한 검색의 요청 횟수의 분포는 의뢰자들의 '최대 한번 가져오는' (Fetch-at-most-once) 성질 때문에 지프(Zipf) 분포에서 벗어나 있다[19]. '최대 한번 가져오는' 성질을 가지는 실측된 부하[19]는 지수분포의 형태를 가지므로, 검색 순위 z인 ($1 \le z \le N$) 질의가 발생하는 빈도수의 분포를 다음과 같이 모형화 할 수 있다.

$$g(z) = e^{-w \cdot z} \tag{8}$$

여기서 상수 w는 정규화를 위한 것이고, 그 값으로 0.001을 사용한다. N은 피어투피어 시스템에 저장된 파일들의 총 수 이고, 연구 [19]에 근거하여, 그 값으로 20000을 사용한다.

그리고, 사용자에 의해 검색이 요청된 파일들의 모든 가능한 순위들의 집합인 표본 공간에 규정된 함수로써 확률변수 X를 규정하자. 함수 $g(z)$를 그것의 확률 밀도를 가지고 정규화하면, 확률변수 X에 대한 확률 밀도 함수를 다음과 같이 구할 수 있다.

$$P(X = x) = e^{-w \cdot x} / \sum_{z=1}^{N} g(z) \tag{9}$$

확률변수 X에 대한 이산 확률밀도 함수는 순위 x 의 파일을 찾기 위한 질의가 일어날 확률을 표현한다. 식 (9)는 캐시 시스템을 이용하면 멀티미디어 파일들을 탐색하는 것에 대하여 피어투피어 시스템의 성능을 상당히 개선할 수 있음을 암시한다.

다음으로, 사용자에 의해 검색이 요청된 파일들의 모든 가능한 순위들의 집합인 표본 공간에 규정된 함수로써 연속 구간 [0,1] 사이의 실수 값을 가지는, 확률 변수 Y를 정의하자. 이러한 확률 변수 Y를 확률 변수 X로부터 다음과 같이 구성할 수 있다.

$$Y = \frac{N - X}{N} \tag{10}$$

우리가 캐쉬를 사용한다는 가정하면, 실측한 작업부하 [19]로부터 확률변수 Y를 식 (10)과 같이 정의할 수 있다. 확률변수 Y는 순위 x의 파일에 관한 질의가 무위도식자가 아닌 노드에서 검색에 성공할 확률을, 즉 적중률을, 표현한다.

위의 식 (9)와 (10)을 사용하여, 수치적 방법으로, 무위도식자(Freeloader)가 아닌 노드들의 적중률의 기대치를 다음과 같이 구할 수 있다.

$$E(Y) = \sum_{x=1}^{N} Y \cdot P(X = x) = \sum_{x=1}^{N} \frac{(N - x)e^{-w \cdot x}}{N \sum_{z} g(z)}$$

$$= \sum_{x=1}^{N} \frac{(N - x)g(x)}{N \sum_{z} g(z)} = 0.949975 \tag{11}$$

이제 [3,20]의 결과를 바탕으로, 첫 통과 시간(Average first passage time)으로서, 검색에 필요한 홉수인 검색 시간을 얻는다. 검색이 성공하기 위해 요구되는 홉 수들의 집합인 표본 공간에 규정된 함수로써 확률변수 H를 규정하자. 기호 h로 홉 수를 나타내고, h-1 개의 노드들을 방문한 후에도 검색에 실패한 상태를 표현하기 위해서 상태 F_{h-1}를 사용하자. h 개의 노드들을 방문한 후에 검색에 성공한 상태를 상태 S로 나타내자.

제안한 프로토콜에 의해서 만들어진 위상이 적중률의 관점에서 볼 때 부분 순서를 가지기 때문에, 메시지의 홉 수가 증가하는 동안 여행하는 질의 메시지는 이전 노드들보다 더 높은 적중률을 가지고 있는 노드를 방문하게 된다. 만약 우리들이 검색을 하여 한 노드에서 목표 파일을 발견할 확률을 표현하기 위해서 기호 a를 사용한다면, 상태 S 와 상태 F_{h-1}의 차등 방정식들은 다음과 같다.

$$S = a \cdot F_{h-1} \tag{12}$$

$$F_{h-1} = (1 - a) \cdot F_{h-2} \tag{13}$$

두 식들을 연립하여 풀면 다음과 같다.

$$S = a \cdot (1 - a) \cdot F_{h-2} = a \cdot (1 - a)^2 \cdot F_{h-3} = a \cdot (1 - a)^{h-1} \tag{14}$$

식 (14)의 a를 $E(Y)$로 대치하면, 확률변수 H에 대한 확률 밀도 함수를 다음과 같은 일종의 기하분포(Geometric distribution)로 구할 수 있다.

$$P(H = h) = E(Y)(1 - E(Y))^{h-1} \tag{15}$$

$\sum_{h=1}^{5} P(H = h) = 1$ 이며, 이는 수치 계산적으로 구할 수 있다. 따라서, 제안한 프로토콜에 의해서 만들어진 위상에서 성공적인 검색을 위해 요구되는 최대 홉수는 확률적으로 5임을 알 수 있다.

처음 다른 노드에 탐색을 요청하는 것은 검색을 위해

서 필수적 행동이므로, 일반성을 잃지 않으면서, 탐색에 필요한 홉 수에서 제외할 수 있다. 따라서, 임의적 1-보행자(Random 1-walker)를 사용하는 경우, 검색의 성공을 위해 요구되는 홉 수의 기대치 $E(H)$를 다음과 같이 구할 수 있다.

$$E(H)=\sum_{h=1}^{\infty}(h-1)\cdot P(H=h)=\frac{1-E(Y)}{E(Y)}=0.05266 \quad (16)$$

추가하여, 3.3.3절에서 제시한 검색 프로토콜에 의해서, 각 노드가 $e^{w\cdot hit\_ratio}$ 만큼의 수의 질의 메시지들을 보낸다면, 홉 수의 기대치는 $E(H)/e^{w\cdot E(Y)}=0.006037$로 줄어든다. 각 노드가 하나의 질의 메시지를 가장 높은 적중률을 가지는 노드로 보내는 최선의 1-보행자(Best 1-walker)를 사용하는 경우에도 성공적인 검색에 필요한 홉 수의 기대치는 식 (16)의 결과와 동일하다. 제안한 연쇄반응적 검색에서 다중 전송되는 질의 메시지들의 적중 사건은 독립사건이고, 메시지들 각각의 적중률은 최선의 1-보행자의 그것보다 작기 때문이다.

제안된 프로토콜에 따라서, 모든 무위도식자 노드들은 서비스를 제공하는 노드들과, 다시 말해 비무위도식자(non-freeloader) 노드들과 각각 연결을 갖는다. 비구조적 피어투 피어 시스템의 노드의 85 퍼센트가 무위도식자이다[6]. 정리 4에 따라서, 우리들은 재결정 위상의 비무위도식자 노드들이 제공하는 연결들의 총 수를 다음과 같이 추정할 수 있다.

$$M\cdot 0.15\cdot e^{w\cdot E(Y)}=M\cdot 0.15\cdot e^{2.28\cdot 0.949975}=1.2478\times M \quad (17)$$

여기서 상수 M은 피어투피어 망을 구성하는 노드들의 총수를 나타낸다.

무위도식자 노드들이 가지는 연결들의 총수는 다음과 같이 추정할 수 있다.

$$M\cdot 0.85\cdot e^{w\cdot E(Y_{freeloader})}=M\cdot 0.85\cdot e^{0}=0.85\times M \quad (18)$$

비무위도식자 노드들의 연결들의 총 수는 무위도식자의 그것의 1.468배이다. 이것은 모든 무위소식자를 비무위도식자들에게 연결하는데 충분하게 보인다.

만약 연결들의 수가 부족하면, 제안된 프로토콜의 상수 w를 증가시켜서 더 많은 연결들을 만들 수 있다. 만약 비무위도식자들이 무위도식자들을 위한 충분한 수의 연결들을 제공한다면, 무위도식자들은 결국 비무위도식자들과 연결될 것이다. 따라서, 일반성의 손실 없이, 질의 메시지가 한 홉에 무위도식 노드를 벗어난다고 이야기 할 수 있다.

4.2 모의 실험

신청된 프로토콜들의 특징과 성능을 분석하기 위하여, 우리들은 NS-2위에 구현되었던 Gnutella 시뮬레이터[21]을 수정함으로써 제시된 프로토콜들을 실현한다. 연구 [19]를 기반으로, 모의실험 파라미터들을 위해 표 1에 제시된 바와 같은 값들을 사용한다.

표 1 모의실험 파라미터들

Symbol	Meaning	Value
M	Number of peers	1000
N	Number of files	20000
$P(X=x)$	Probability that user request a file of popularity rank x	Zipf(2.28)
-	Client arrival rate	Uniform
-	Freeloader rate	0.85

라우터들로 구성된 인터넷 위상을 생성하기 위해, 우리들은 위상 생성기 BRITE[22]를 사용한다. 위상생성을 위해 Barabasi와 Albert에 의해 제안된 제곱 법칙 망 모형[23]을 실현한 Barabasi-Albert 망 모형을 사용한다. 우리들은 다음과 같은 파라미터들을 사용한다. 라우터의 수는 1000 대 이다. 노드들은 무거운 꼬리(Heavy tailed) 분포에 의해서 놓여진다. 각 새로운 노드가 연결된 인근 노드들의 수는 2 이다. 대역폭 분포는 최소 10Mbps 그리고 최대 1024Mbps를 지닌 무거운 꼬리 분포이다.

제안된 프로토콜들을 평가하기 위해서, 모의실험기에 구현된 제안된 프로토콜을 사용하여 재결정 위상을 만들고, 또한 보수적인 비교를 하기 위해 원래의 그누텔러 프로토콜을 사용하여 각 노드가 재결정 위상의 노드 당 평균 연결 차수의 두 배만큼의 연결들을 가지는 위상을 생성한다. 그리고, 재결정 위상에서 연쇄반응적 검색 프로토콜과 최선의 1-보행자 검색 프로토콜을 수행 한다. 그리고, 원래의 그누텔러 프로토콜로 생성된 위상에서 플러딩 검색 프로토콜과 임의적 1-보행자 검색 프로토콜을 수행 한다. 각 모의 실험을 위해 1000 초의 모의 실험 시간을 사용한다. 임의로 선택된 서로 다른 3개의 시나리오들을 사용하여 얻은 총 3번의 모의 실험들의 결과들을 평균하여, 본 절에서 제시된 모든 결과들을 얻었다.

4.3 비교 분석

정량적인 비교를 위해, 우리들은 탐색 시간과 검색 비용에 관한 모의 실험 결과들과 분석 결과들을 제시한다. 또한, 모의실험 결과로서 검색 성공률과 평균 응답시간을 제시한다. 그림 7에 보인 바와 같이, 재결정 위상에서 최선의 1-보행자를 사용하는 경우에, 검색 성공률을 임의적(random) 1-보행자를 사용하는 원래 그누텔러 보다 10% 개선하며, 플러딩을 사용하는 그누텔러 보다 12% 개선한다. 재결정 위상에서 연쇄 반응적 보행자들을 사용하는 경우에, 검색 성공률을 임의적 1-보행자를 사용하는 경우에도 각각 비교대상에 대해 동일한 백분율 만큼 개선한다. 그렇지만, 재결정 위상에서 제안한 검색 방법들을 사용하는 경우에 절대적 검색 성공률이

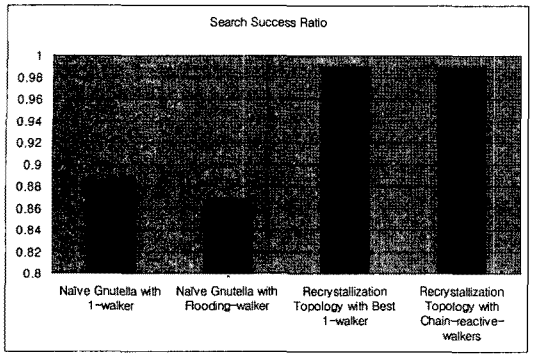

그림 7 제안된 프로토콜 그리고 원래의 그누텔러 프로토콜에 의해 구성된 위상에서 다양한 검색 전략들의 검색 성공률들

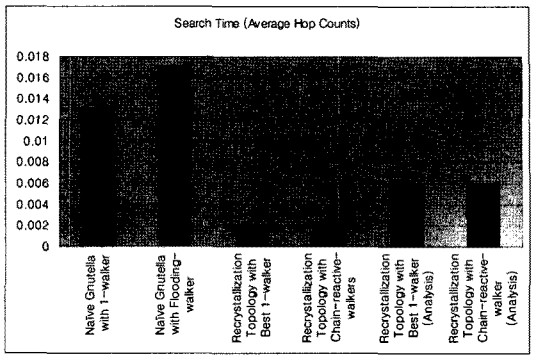

그림 9 제안된 프로토콜 그리고 원래의 그누텔러 프로토콜에 의해 구성된 위상에서 다양한 검색 전략들의 검색 시간들

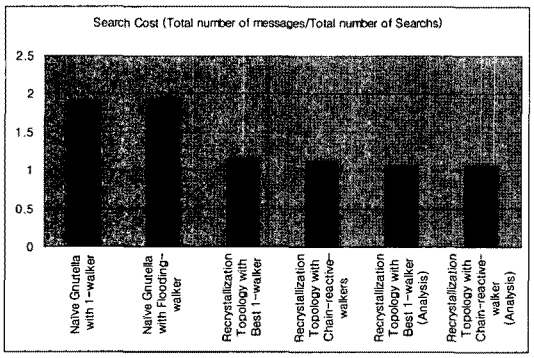

그림 8 제안된 프로토콜 그리고 원래의 그누텔러 프로토콜에 의해 구성된 위상에서 다양한 검색 전략들의 검색 비용들

99 퍼센트인 것은 주목할 만하다.

그림 8에 보인 바와 같이, 재결정 위상에서 최선의 1-보행자를 사용하는 경우에, 임의적 1-보행자를 사용하는 원래 그누텔러 보다 검색 비용 즉 하나의 성공적인 검색을 위해 만들어지는 메시지들의 수를 40%만큼 절약하며, 플러딩을 사용하는 그누텔러 보다 41% 만큼 절약한다. 재결정 위상에서 연쇄 반응적 보행자들을 사용하는 경우에, 임의적 1-보행자를 사용하는 원래 그누텔러 보다 검색 비용을 41%만큼 절약하며, 플러딩을 사용하는 원래의 그누텔러 보다 43%만큼 절약한다. 재결정 위상에서 연쇄 반응적 보행자들을 사용하는 경우에 하나의 성공적인 검색을 위해서 요구되는 메세지들의 수의 평균이 원래 사용자들에게서 발생한 메시지들의 수의 단지 1.12 배임에 유의하자. 모의실험 결과들은 분석 결과들과 매우 유사하다.

그림 9에 보인 바와 같이, 재결정 위상에서 최선의 1-보행자를 사용하는 경우에, 임의적 1-보행자를 사용하는 원래 그누텔러 보다 검색 시간을, 다시말해 성공적 검색에 필요한 홉 수를, 87%만큼 절약하며, 플러딩을 사용하는 그누텔러 보다 90%만큼 절약한다. 재결정 위상에서 연쇄 반응적 보행자들을 사용하는 경우에, 임의적 1-보행자를 사용하는 원래 그누텔러 보다 검색 홉 수를 82%만큼 절약하며, 플러딩을 사용하는 원래의 그누텔러 보다 86%만큼 절약한다. 재결정 위상에서 연쇄 반응적 보행자들을 사용하는 경우에 성공적인 검색에 필요한 홉수가 원래 그누텔러에서 임의적 1-보행자를 사용하는 경우의 그것의 18%에 불과 한 것은 주목할 만 하다. 그림 9의 모의실험 결과들과 분석결과는 약간의 차이를 보이는데, 이는 모의실험에서 검색 요청의 모형으로 일반적으로 사용되는 Zipf 분포를 사용하였고, 분석에서는 유사하지만 완전히 같지는 않은 제시한 확률 분포를 사용하였기 때문이다. 모의실험에서 사용된 Zipf 분포의 상위 계층의 파일의 검색 빈도수가 분석에서 사용된 분포의 그것보다 크기 때문에, 발견될 확률이 높은 상위 계층들의 파일들이 상대적으로 많이 요청된 모의실험에서의 탐색시간이 평균적으로 좀더 작게 나타난 것으로 보여진다.

그림 10에 보인 바와 같이, 재결정 위상에서 최선의 1-보행자를 사용하는 경우에, 임의적 1-보행자를 사용하는 원래 그누텔러 보다 성공적인 검색을 위한 평균 응답시간을 93%만큼 절약하며, 플러딩을 사용하는 원래의 그누텔러 보다도 93% 만큼 절약한다. 재결정 위상에서 연쇄 반응적 보행자들을 사용하는 경우에, 임의적 1-보행자를 사용하는 원래의 그누텔러 보다 평균 응답시간을 93%만큼 절약하며, 플러딩을 사용하는 그누텔러에 비해 99% 만큼 절약한다. 재결정 위상에서 연쇄 반응적 보행자들을 사용하는 경우에 평균 응답시간은 원래 그누텔러에서의 임의적 1-보행자의 그것의 7%에 불과한

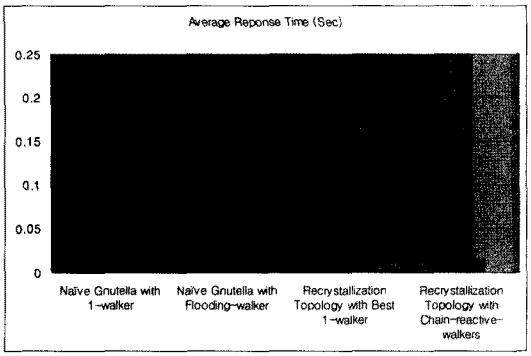

그림 10 제안된 프로토콜 그리고 원래의 그누텔러 프로토콜에 의해 구성된 위상에서 다양한 검색 전략들의 평균 응답시간들

것은 주목할 만하다. 결과적으로, 연쇄반응적 검색과 구성된 척도-없는 거듭제곱법칙 위상을 사용함으로써, 우리들은 99 퍼센트까지 검색 성공 비를 증가시키면서 탐색 시간과 검색 비용을 절약할 수 있다.

4.4 부하분산 및 확장성

각 노드의 차수 k인 연결들로부터 질의 메시지를 받아서 i 개의 검색 성공 할 확률은, 각 질의 메시지의 검색 성공이 서로 독립적이며, 성공할 확률이 hit_ratio 이라면, 일반적인 이항 분포이다. 따라서, 이 이항 분포의 기대값 즉 이 노드에서 검색에 성공하는 입력 메시지들의 기대값은 다음과 같다.

$$E[\text{검색에성공한메시지수}] = \sum_{i=0}^{k} i \cdot \binom{k}{i} (hit\_ratio)^i (1-hit\_ratio)^{k-i} = k \cdot hit\_ratio \quad (19)$$

따라서, 추가로 메시지를 방송하게 되는, 검색에 실패한 메시지들의 수의 기대치는 다음과 같이 구할 수 있다.

$$E[\text{검색에실패한메시지수}] = k - k \cdot hit\_ratio \approx k - k \cdot E(Y) = 0.050025 \cdot k \quad (20)$$

따라서 검색에 실패한 하나의 메시지에 대해서, k 개의 메시지들이 전파되므로, 다음 홉으로 전달되는 패킷의 수는 총 $0.050025 \cdot k^2$개이다. 따라서, 각 노드의 차수의 제곱에 비례하나, 차수는 최대 차수는 $\lfloor e^{2.28} \rfloor = 9$ 이고, 이 경우 최대 4.052개의 패킷들이 방송되므로, 대규모 검색에서 병목현상은 발생하지 않을 것으로 보여진다. 따라서, 확장성이 있다.

5. 결 론

본 논문에서 피어들 간에 다면평가(Multisource feedback)를 사용하여 척도-없는 망을 구성하는 완전히 분산된 위상 제어 프로토콜을 제시하였다. 구성된 위상은 노드들의 검색 능력 간에 부분 순서가 존재하는 척도-없는 망이다. 각 노드의 연결들의 수는 자신의 검색 능력에 지수적으로 비례한다. 따라서 구성된 망의 노드들의 차수 서열(Degree sequence)의 분포는 거듭제곱 법칙에 따르며, 결과적으로, Cohen의 연구에서 제시된 척도-없는 망의 구성 과정 모형으로 설명되듯이, 각 피어는 자신과 유사한 검색 능력을 가지는 다른 피어들과 연결하여 연합한다. 구성된 위상이 검색 능력에 전역적으로 최적인 망을 구성하는 사실을 설명하기 위해서, 본 논문에서 제시된 프로토콜과 볼츠만 기계와의 연결을 밝히고, 위상 최적화 문제를 볼츠만 기계 관점에서 모형화했다.

제안한 프로토콜은, 다면평가 결과를 전파하는 메시지를 보내기 위해서, 자신의 능력에 지수적으로 비례하는 TTL을 사용하는 제어된 다중 전송방식을 사용함으로써, 효과적으로 다면평가의 결과를 전파하면서도, 효율적으로 척도-없는 망의 위상을 구성한다. 또한, 구성된 위상의 특성들을 활용하는 효율적인 검색 프로토콜을 제안하였다. 검색 성공률은 99 퍼센트 이다. 제시된 제어된 다중 전송 방식의 검색은 방송을 사용하는 방식보다 질의 메시지 수를 43 퍼센트만큼 줄인다. 제안된 검색 프로토콜이 원천 노드들에서 만들어진 원 질의 메시지의 11 퍼센트에 불과한 추가 메시지들을 만든다는 사실에 유의할 필요가 있다.

제안된 프로토콜에 의해서 구성되는 척도-없는 위상에서 존재하는 부분순서가 때문에, 한 질의 메시지가 노드를 방문하면, 그 노드는 직전에 방문되었던 노드보다 더 높은 적중률을 가지고 있다. 더 나아가, 무위도식(Freeloader) 노드로부터 보내진 질의 메시지는 한 홉의 전달에 의해 높은 적중률을 가지고 있는 노드로 탈출할 수 있고, 그것은 다시는 무위도식 노드를 방문하지 않는다. 이처럼 검색은 한정된 탐색시간 안에 이루어질 수 있다. 이것은 또한 검색 성공율을 개선한다.

우리들은 제안된 위상 제어 프로토콜의 거동을 재결정 현상의 재결정 알갱이의 형성과정을 가지고 은유적으로 설명할 수 있다. 재결정 현상은 풀림 동안 금속성 물질에서 관찰된다. 퍼텐셜이 높은 분자들이 멀리 에너지를 전달하며, 퍼텐셜이 높은 분자들과 재결정 핵을 형성한다. 유사하게, 적중률이 높은 노드들이 멀리 메시지를 보내며, 적중률이 높은 노드들과 결집하여 재결정 핵을 형성한다.

제시된 프로토콜에 의해 구성된 위상에 3-코어연산(3-core operation)을 적용하면, 몇 개의 분할들을 발견할 수 있었다. 1000개의 피어들로 구성된 망을 모의실험한 경우에 구성된 위상의 노드들의 대부분이 2개의 분

할들을 구성한 것을 확인하였다. 분할의 노드들을 배치하기 위해서 Kamada-Kawai 알고리즘을 사용하면, 노드들은 동심원의 형태로 배치됨을 볼 수 있었다. 이러한 결과는 Cohen이 제시한 척도-없는 망의 구성과정 모형과 일관성이 있다.

제안된 프로토콜 그리고 원래의 그누텔러 프로토콜에 의해 구성된 위상에서 다양한 검색 전략들의 정량적 비교를 제공하기 위하여, 우리들은 분석적인 결과들과 모의 실험 결과들을 제공한다. 재결정 위상에서 최선의 1-보행자를 사용하는 경우에, 임의적 1-보행자를 사용하는 원래 그누텔러 보다 검색 시간을, 다시 말해 성공적 검색에 필요한 홉 수를, 87%만큼 절약하며, 플러딩을 사용하는 그누텔러 보다 90% 만큼 절약한다. 재결정 위상에서 연쇄 반응적 보행자들을 사용하는 경우에, 임의적 1-보행자를 사용하는 원래 그누텔러 보다 검색 홉 수를 82%만큼 절약하며, 플러딩을 사용하는 원래의 그누텔러 보다 86%만큼 절약한다. 재결정 위상에서 연쇄 반응적 보행자들을 사용하는 경우에 성공적인 검색에 필요한 홉수가 원래 그누텔러에서 임의적 1-보행자를 사용하는 경우의 그것의 18 %에 불과 한 것은 주목할 만하다.

재결정 위상에서 최선의 1-보행자를 사용하는 경우에, 임의적 1-보행자를 사용하는 원래 그누텔러 보다 검색 비용 즉 하나의 성공적인 검색을 위해 만들어지는 메시지들의 수를 40%만큼 절약하며, 플러딩을 사용하는 그누텔러 보다 41%만큼 절약한다. 재결정 위상에서 연쇄 반응적 보행자들을 사용하는 경우에, 임의적 1-보행자를 사용하는 원래 그누텔러 보다 검색 비용을 41%만큼 절약하며, 플러딩을 사용하는 원래의 그누텔러 보다 43% 만큼 절약한다. 재결정 위상에서 연쇄 반응적 보행자들을 사용하는 경우에 하나의 성공적인 검색을 위해서 필요한 메세지들의 수가 원래 사용자들에게서 발생한 메시지들의 수의 단지 1.12 배임에 유의하자.

참 고 문 헌

[1] R. Matei, A.Iamnitchi, and P.Foster, "Mapping the Gnutella network," *IEEE Internet Computing*, Vol. 6, No. 1, pp. 50-57, 2002.

[2] Lada A. Adamic, Rajan M. Lukose, Amit R. Puniyani, and Bernardo A. Huberman, "Search in power-law networks," *Physical Review E*, Vol. 64, No. 4, pp. 046135, 2001.

[3] Saurabh Tewari and Leonard Kleinrock, "Optimal search performance in unstructured peer-to-peer networks with clustered demands," *IEEE Journal on Selected Areas in Communications*, Vol.25, No. 1, pp. 84-95, 2007.

[4] Alexander Loser, Steffen Staab, and Christoph Tempich, "Semantic social overlay networks," *IEEE Journal on Selected Areas in Communications*, Vol. 25, No. 1, pp. 5-14, 2007.

[5] Xiaoyan Song, and Markus Rettenmayr, "Modeling recrystallization in a material containing fine and coarse particles," *Computational Materials Science*, Available online, 2006.

[6] D. Hughes, G. Coulson, and J. Walkerdine, "Free riding on Gnutella revisited: the bell tolls?," *IEEE Distributed Systems Online*, Vol. 6, No. 6, 2005.

[7] H. Guclu and M. Yuksel, "Scale-Free Overlay Topologies with Hard Cutoffs for Unstructured Peer-to-Peer Networks," *27th International Conference on Distributed Computing Systems (ICDCS '07)*, USA, pp. 32, 2007.

[8] T. Condie, S. Kamvar, and H. Garcia-Molina, "Adaptive peer-to-peer topologies," *Proc. 4th Int. Conf. Peer-to-Peer Comput.*, pp. 53-62, 2004.

[9] H. Tian, S. Zou, W. Wang and S. Cheng, "Constructing efficient peer-to-peer overlay topologies by adaptive connection establishment," *Computer Communications*, Vol. 29, No. 17, pp. 3567-3579, 2006.

[10] R. Cohen and S. Havlin, "Scale-free networks are ultrasmall," *Physical Review Letters*, 90:058701, 2003.

[11] P. Dasgupta, "A Multi-agent Mechanism for Topology Balancing in Unstructured P2P Networks," *Proceedings of the IEEE/WIC/ACM international Conference on intelligent Agent Technology*, IAT 2006, Washington DC, pp. 389-392, 2006.

[12] S. Merugu, S. Srinivasan, and E. Zegura, "Adding structure to unstructured peer-to-peer networks: The use of small-world graphs," *J. Parallel and Distributed Computing*, pp. 142-153, 2005.

[13] S. Amari, K. Kurata, and H. Nagaoka, "Information geometry of Boltzmann machines," *IEEE Transactions on Neural Networks*, Vol.3, No.2, pp. 260-271, 1992.

[14] E. Aarts and J. Korst, *Simulated annealing and Boltzmann machines: a stochastic approach to combinatorial optimization and neural computing*, *Reading Mass.*, New York: John Wiley & Sons, Inc., 1989, pp. 126-177.

[15] M. Fleischer, "Simulated annealing: past, present, and future," *Proceedings of the 1995 Winter Simulation Conference*, pp. 155-161, 3-6 Dec. 1995.

[16] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: overview and conceptual comparison," *ACM Comput. Surv.*, Vol. 35, No. 3, pp. 268-308, 2003.

[17] J. D Vicente, J. Lanchares, and R. Hermida, "Annealing placement by thermodynamic combinatorial optimization," *ACM Trans. Des. Autom. Electron. Syst.*, Vol. 9, No. 3, pp. 310-332, 2004.

[18] C. Kittel and H. Kroemer, *Thermal Physics*, 2nd ed., *Reading Mass.*, New York: W.H. Freeman,

1980, pp. 42-64.
[19] Krishna P. Gummadi, Richard J. Dunn, Stefan Saroiu, Steven D. Gribble, Henry M. Levy, and John Zahorjan, "Measurement, modeling, and analysis of a peer-to-peer file-sharing workload," *Proceedings of the nineteenth ACM symposium on Operating systems principles 2003*, USA, pp. 314-329, October, 2003.
[20] W. Feller, *An Introduction to Probability Theory and Its Applications*, vol. 1, *Reading Mass.*, New York: John Wiley & Sons, Inc., 1950, pp. 344-349.
[21] Q. He, M. Ammar, G. Riley, H. Raj, and R. Fujimoto, "Mapping peer behavior to packet-level details: a framework for packet-level simulation of peer-to-peer systems," *Proceedings of 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer Telecommunications Systems, MASCOTS 2003*, USA, pp. 71-78, October, 2003.
[22] A. Medina, A. Lakhina, I. Matta, and J. Byers, "Brite: An approach to universal topology generation," *Proc. of International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunications Systems (MASCOTS) 2001*, pp. 346-356, 2001.
[23] Reka Albert and Albert-Laszlo Barabasi, "Topology of evolving networks: local events and universality," *Phys. Rev. Lett.*, Vol. 85, No. 24, pp. 5234-5237, 2000.

박 재 현

1988년 2월 중앙대학교 전자계산학과 졸업. 1991년 2월 한국 과학기술원 전산학과 석사. 1995년 8월 한국 과학기술원 전산학과 박사. 1995년 8월~2000년 2월 삼성전자 정보통신 본부 데이타네트웍 개발팀 MPLS/ATM 프로토콜 개발 실무책임. 2000년 3월~2002년 8월 영남대학교 전자정보공학부, 정보통신 전공 교수. 2002년 9월~현재 중앙대학교 컴퓨터공학부 교수. 관심분야는 ATM Switch Arch., Multi-protocol Label Switching System, Routing Protocols, Ad Hoc Networking, Peer-to-Peer System