인피니밴드/RDMA 기반 분산 그래프 데이터 처리 엔진

2세부

KNU 강원대학교

연구 동기

- 그래프 데이터는 복잡한 구조로 인해 단일 노드 시스템에서의 처리 한계 존재
- 그래프의 분산·병렬처리 모델(예: Pregel) 역시 대용량 데이터에서 부하 발생

연구 목표

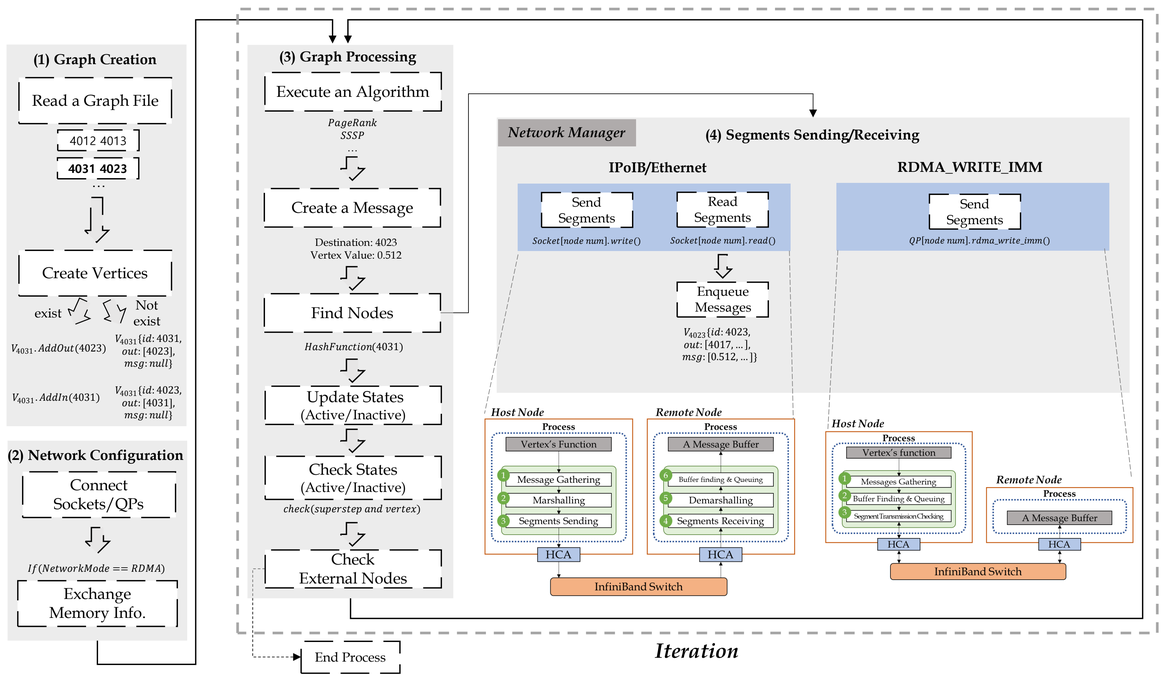
- Pregel 모델에서 발생되는 과다한 네트워크 통신 비용 해결
  - 그래프 처리 프로세스 및 네트워크 통신 과정 경량화를 통한 통신량 감소 방안 연구
- 네트워크 프로토콜 개선으로 전체 처리 성능 향상 가능성 확인
  - Apache Giraph와의 성능 평가를 통한 프로토콜 효율성 입증

관련 기술

- Pregel
  - 구글에서 사용한 vertex-centric 기반 그래프 처리 모델
  - BSP(Bulk Synchronous Parallel)*에서 영향을 받음
  - 대표 적인 Pregel 기반 프레임워크로 Apache Giraph가 있음
- 인피니밴드(InfiniBand)
  - 높은 대역폭과 낮은 전송 지연시간을 보장하는 고성능 통신 장비
  - IPoIB(IP over InfiniBand): 인피니밴드 H/W 상에서 이더넷과 같이 TCP/IP 계층을 통해 네트워크 통신 수행
  - RDMA(Remote Direct Memory Access): 호스트 메모리에서 원격지 메모리로 직접 데이터를 전송하는 기능

PIGraph의 전체 동작 구조도

- PIGraph (Pregel and InfiniBand-based graph processing engine)
  - Pregel 모델에 인피니밴드의 두 가지 프로토콜(IPoIB, RDMA)을 적용한 그래프 처리 엔진
  - RDMA 모델 적용을 위한 그래프 처리 및 통신 과정 개선
- PIGraph 전체 동작 구조
  - 총 네 단계의 처리 과정(그래프 생성, 네트워크 설정, 그래프 처리, 세그먼트 송수신)
  - 프로토콜에 따라 다른 네트워크 설정 지원
  - 그래프 처리 알고리즘(PageRank, SSSP 등)은 병렬로 수행



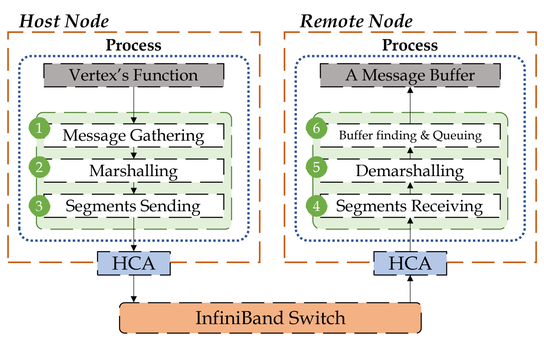
PIGraph의 주요 동작 과정

① 그래프 생성
- 방향성 그래프 정보가 포함된 파일 데이터 입력
- 정점, 간선 정보가 포함된 방향성 그래프 데이터 파일에서 출발 정점을 id로 하는 vertex 클래스 생성

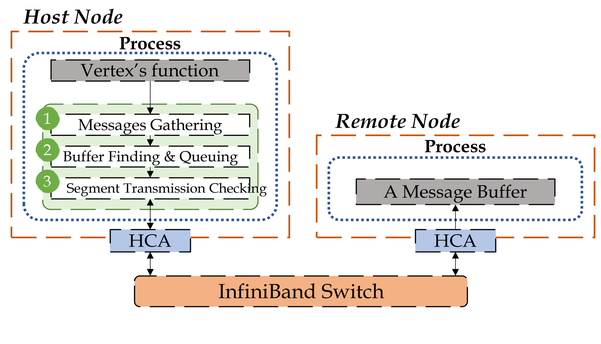
② 네트워크 설정
- IPoIB: 클러스터의 모든 노드 정보를 기반으로 소켓 연결
- RDMA: 메타 정보 전달을 위한 소켓 연결, RDMA 수행을 위한 queue pair 연결

③ 송신 전 그래프 처리
- 각 정점에 대한 그래프 질의 알고리즘 수행 후 결과 메시지 생성
- 목적지 노드의 소켓/queue pair 검색

④ 세그먼트 송수신
- 호스트 노드에서 원격 노드로 세그먼트를 송수신
- IPoIB와 RDMA 환경에서 발생하는 송수신 과정이 다르게 진행



<IPoIB의 세그먼트 송수신>

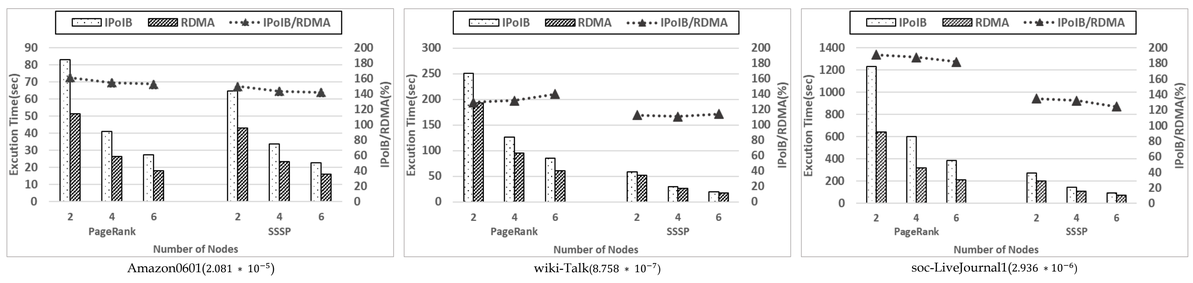


<RDMA의 세그먼트 송수신>

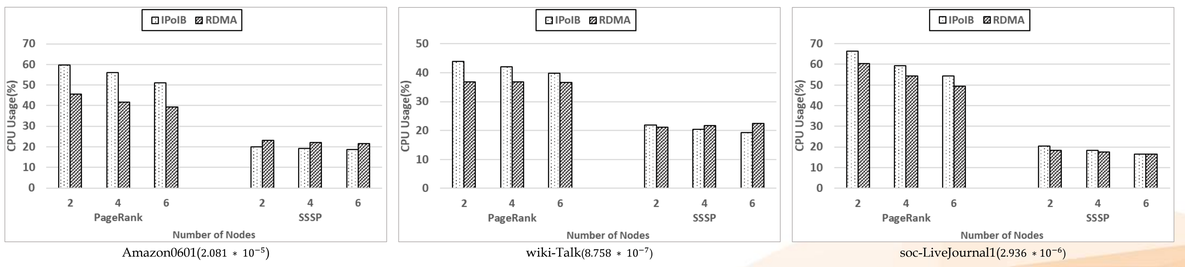
⑤ 수신 후 그래프 처리
- 각 정점의 처리 상태를 확인/갱신
- 전체 노드의 정점 상태 공유/확인 후 활성화된 정점이 없다면 현재 프로세스 종료

성능 평가

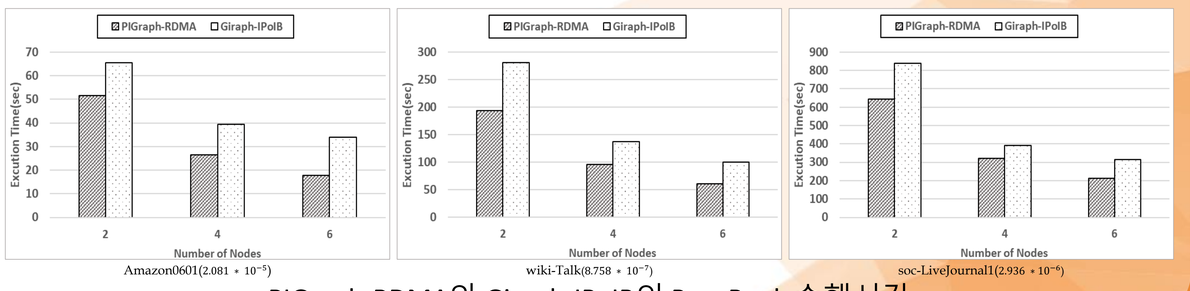
- Apache Giraph와의 성능 평가 결과, RDMA 프로토콜 사용 시 최대 190% 이상 처리 시간 감소



< PIGraph-IPoIB와 PIGraph-RDMA의 알고리즘 수행시간>



<PIGraph-IPoIB와 PIGraph-RDMA의 CPU 사용률>



<PIGraph-RDMA와 Giraph-IPoIB의 PageRank 수행시간>