Semantics-based Distributed I/O for mpiBLAST

P. Balaji[α], W. Feng[β], J. Archuleta[β], H. Lin[δ], R. Kettimuthu[α], R. Thakur[α] and X. Ma[δ]

[α] Argonne National Laboratory

[β] Virginia Tech

[δ] North Carolina State University

Traditional Distributed I/O

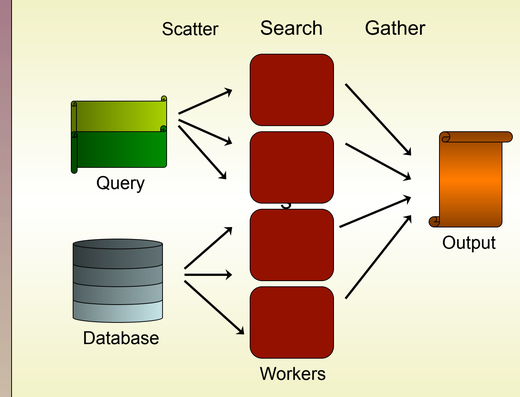
Issues with Distributed I/O

- High Latency
  - Synchronization operations heavily effected
- Low Bandwidth
  - High-bandwidth links are not accessible to everyone
- Data Encryption
  - Distributed I/O over the Internet might need to be encrypted in some environments
- And yet distributed I/O is essential
  - Large-scale computations might require resources that are not available at a single site
  - Scientists may need to access remote large-scale supercomputers which are not available locally

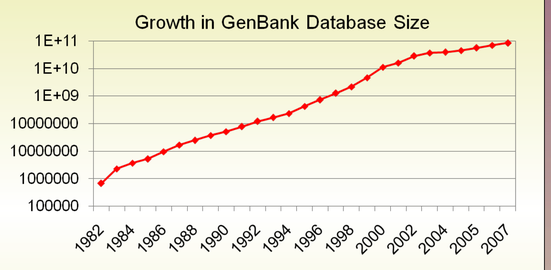
NSF TeraGrid

Sequence Search with mpiBLAST

mpiBLAST Working Model



Distributed I/O in mpiBLAST

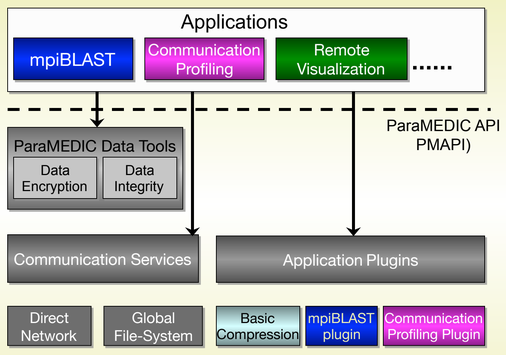


Estimated Output of an All-to-All NT Search

Query Size (KB)	0-5	5-50	50-150	150-200	200-500	>500	Total
Number of Queries	3,305,170	87,506	25,920	26,524	9,592	248	3,455,000
Estimated Output (GB)	1,139	593	23,555	3,995	?	?	>29,282

ParaMEDIC Framework

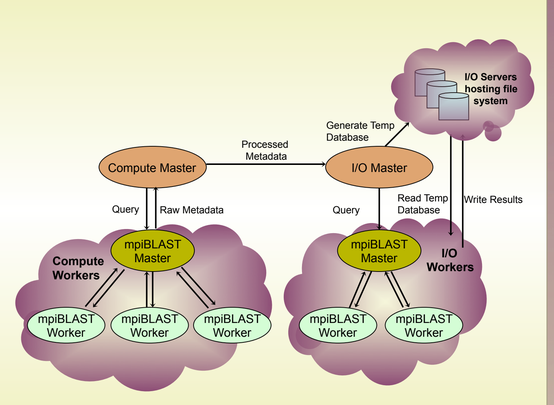
ParaMEDIC Architecture



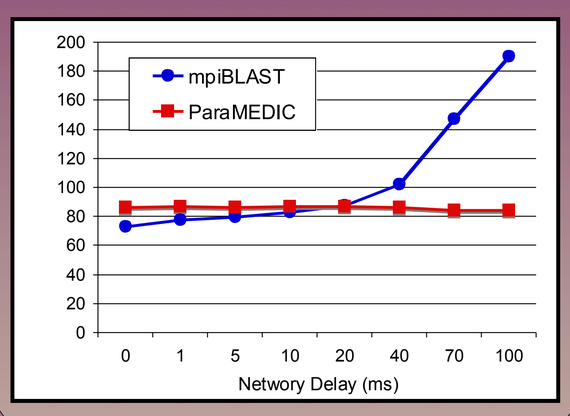
Understanding ParaMEDIC

- Primary Idea:
  - Data transformation, but not at the byte-level (unlike compression techniques)
  - Application specifies the appropriate description of the data → allows ParaMEDIC to process output as high-level objects, and not a stream of bytes
- With respect to mpiBLAST
  - Overall output is just a concatenation of matches and other descriptive information for each query sequence
    - Match scores, alignment information, etc.
  - Finding the database matches for each sequence is the compute intensive portion → needs to be stored
    - Rest can be discarded and regenerated

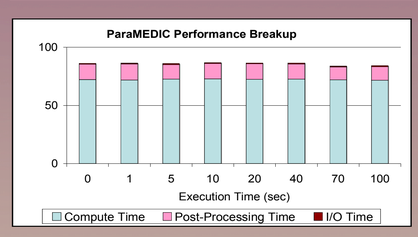
ParaMEDIC-powered mpiBLAST



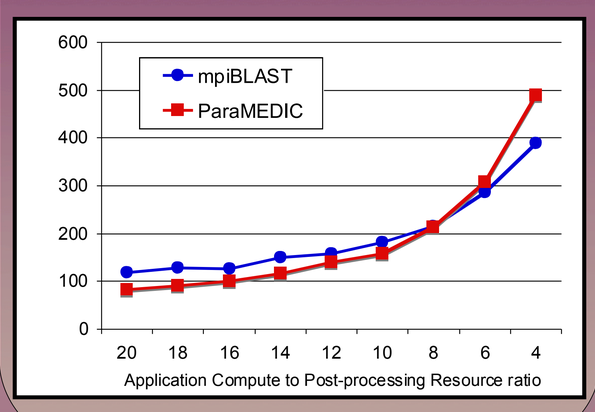
Impact of Network Latency



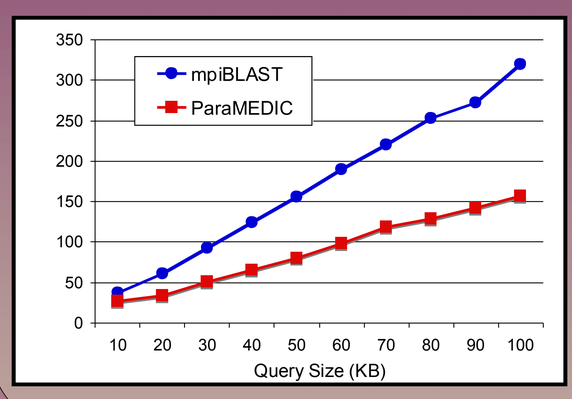
Network Delay: Performance Breakup



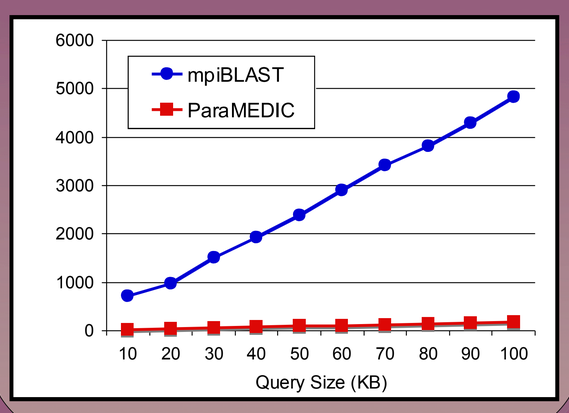
Trading Computation and IO



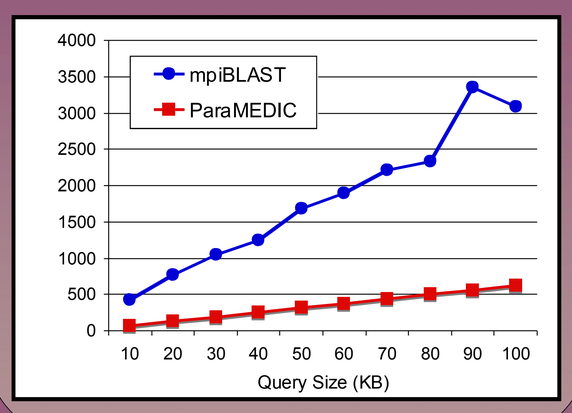
Impact of Encrypted File-Systems



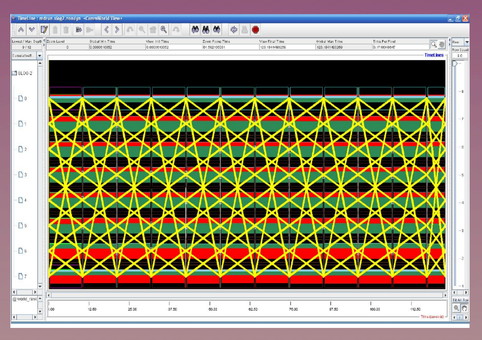
Argonne-VT Distributed Setup



Performance on TeraGrid



Other Applications: MPE Communication Profiler



- Scientific applications have periodic communication profiles
- ParaMEDIC uses FFT to find periodicity and process output
  - Preliminary results show 2-5X reduction in output

Conclusion

- Distributed I/O is a necessary evil
  - Large applications need resources from multiple centers
  - Scientists might need to use remote large-scale resources
- Impacted by several issues
  - High Latency, Low Bandwidth, Encryption requirements
- We propose the ParaMEDIC framework
  - Semantics-based Distributed I/O mechanism
  - ParaMEDIC uses application-specific plugins to "understand" what the output data means and process it
  - Output data converted to application-specific metadata, transported, and then converted back
- Trades a small amount of additional computation for potentially large benefits in I/O