A Parallel Python Implementation of BLAST+ (PPIB) for Characterization of Complex Microbial Consortia

Amina Jackson[1,2], William Conner Horne[1], Daniel Beall[1], Kenneth Jiang[1], William Judson Hervey, IV[1]

[1]Center for Bio/Molecular Science & Engineering, Naval Research Laboratory (NRL-DC), Washington, DC, USA [2]Jerome and Isabella Karles Distinguished Scholar, NRL-DC

Overview

• Technological advancements in analytical instrumentation have made routine tasks among the "-omics" sciences increasingly computationally demanding.

• High performance computing (HPC) offers computing power to handle the most computer intensive and complex tasks in "-omics" like sequence database-searches.

• Here, we demonstrate BLAST+ deployed on Xeon Host with parallel python implementation completes BLAST on real-world complex microbiomes, but does not complete within the allocated wall time for BLAST+ as configured out of the box

Introduction

Leveraging the computational power of HPC among the "-omics" sciences of genomics, transcriptomics, and proteomics is highly desirable to offset a growing "big data" problem, yet application deployment on HPC architectures poses significant challenges to this multidisciplinary research area. The emergence of analytical platforms that rapidly acquire biomolecular sequence data, such as third-generation genome sequencers and high mass accuracy mass spectrometers, have caused an explosive growth among biomolecular sequence data. The NCBI's BLAST+, a tool essential to "-omics" workflows, infers putative functions of biomolecular sequences via similarity to large-scale reference sequences of known origin. Though disparate BLAST+ implementations have been updated to harness multiple threading on large-memory servers and some forms of parallelism, we have found such versions are not regularly maintained or updated, often require substantial modifications of HPC environments, and are not tractable for application to our large-scale microbial consortia (or microbiomes). Here, we compare BLAST+ relative to a ***P****arallel* ***P****ython* ***I****mplementation* of ***B****LAST+ (or **PPIB**) on 36, 108 and 216 cores of an HPC system. Four disparate biological sequences representative of environmental microbiomes were used as input for both BLAST+ and PPIB and compared to the NCBI non-redundant (nr) protein reference. BLAST+ was executed for putative functional assignment but failed to efficiently utilize resource allocations to scale and on real-world microbiomes failed to complete processing within resource allocation limits. The same execution with PPIB demonstrated efficiency by scaling based on resource allocations and real-world microbiomes completed putative functional assignments before exhausting resource allocation limits.

Methods: Computational Approach

Job Submission and Processing on HPC System "Thunder" (SGI Ice X)

A. BLAST+ PBS Job Process
PBS loads input-file on Xeon Host

Job Processing characteristics:

- Without specification, cores perform the same tasks

BLASTnr Reference Database

- Input is sequentially processed searching single queries to completion.

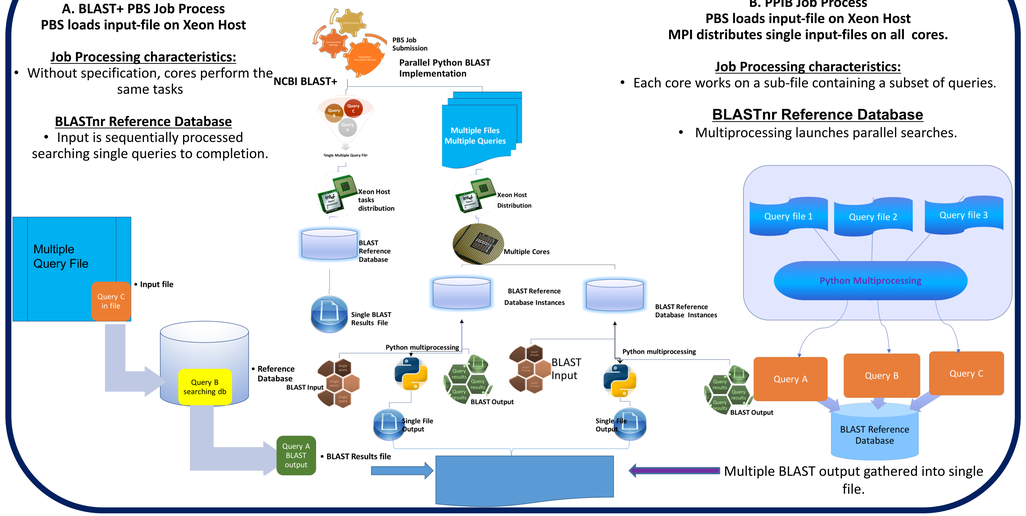
B. PPIB Job Process
PBS loads input-file on Xeon Host
MPI distributes single input-files on all cores.

Job Processing characteristics:

- Each cores works on a sub-file containing a subset of queries.

BLASTnr Reference Database

- Multiprocessing launches parallel searches.



Methods: Input Data from Environmental Microbial Consortia

A. Microbial Consortia (or Microbiomes) Characterized via "-omics" Approaches

1. Microbial Fuel Cell Biocathode MCL (Glaven Laboratory, NRL-DC)

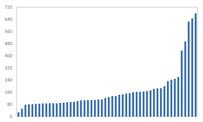
2. Ship Hull Biofilm (Vora Laboratory, NRL-DC)

Systems biology approaches were previously applied to determine the metagenomes and metaproteomes of the Biocathode-MCL microbiome (Wang *et al. Applied Env. Micro.* 2014; Leary, Hervey, *et al. PROTEOMICS*. 2015) and Ship Hull Biofilm (SHB) N1 microbiome (Leary *et al Anal. Chem.* 2012; Leary *et al. Biofouling* 2014).

These complex environmental microbiomes posed a significant computational challenge for putative functional inference via BLAST+ via multithreading on large memory servers (eg. no HPC). Thus, this is the motivation behind the design and implementation of ***PPIB***: to make putative functional assignments by sequence homology via BLAST+ tractable on HPC systems.

B. Distribution of Input Sequence File Lengths (number of amino acids)

1. Test Input of Long Sequence Entries (n=52)

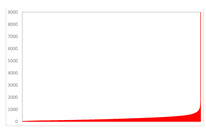


Long Sequences: 52 queries of SHB data set that had greater than 667 amino acid resides.

2. Test Input of Short Sequence Entries (n=200)

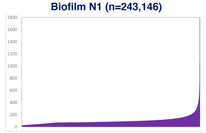
Short Sequences: 200 queries of SHB data set that had less than 667 amino acid resides.

3. Predicted Metaproteome of Biocathode-MCL (n=79,765)



Real-world Microbiomes:
Biocathode-MCL
2,807 of 79765 were long sequences

4. Predicted Metaproteome of Ship Hull Biofilm N1 (n=243,146)



Real-world Microbiomes:
SHB:
305 of 243146 queries were long sequences.

C. PPIB Implementation

Input: A single input file is divided into as many sub files as the are cores.

Process: Each core solves a different part of the problem in parallel greatly reducing overall wall time

i. Python MPI function Scatter distributes each file per core.
ii. Python Multiprocessing creates multiple database instances.

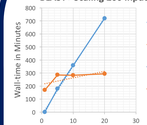
Implication: A job allocated 36 cores can have up-to 100*36 = 3600 queries processes searching the database in parallel.

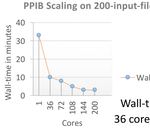
Output:

i. Multiple parallel searches generate multiple output files
ii. Python MPI function Gather gathers BLAST output into single output file.

Results: Scaling and Performance Comparisons

A. Scaling Comparisons between BLAST+ and PPIB

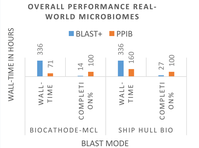




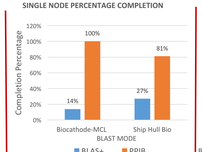
BLAST+ 200-Input-file
Wall-time increase as cores increase.
180 cores, the wall-time was 286 minutes while on a single core the wall-time was 170 minutes

PPIB 200-query Data Test
Wall-time decreased as the number of cores increased.
36 cores took 10 minutes while 108 cores took 5 minutes.

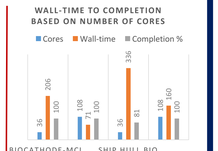
B. Wall-Time performance between BLAST+ and PPIB under wall-time maximum of 336 hours



Biocathode-MCL: Of 79,765 queries BLAST+ processed 11,490 in 336 hours. PPIB completed the job in 71 hours .
SHB: Of 243,146 queries, BLAST+ processed 65759 after 336 hours. PPIB completed the job in 160 hours.



Biocathode-MCL: BLAST+ processed 14% queries after 336 hours. PPIB completed job processing in 209 hours.
SHB: BLAST+ processed 27% queries at 336 hours. PPIB processed 81% in 336 hours.



Biocathode-MCL: Core increase from 36 cores to 108 cores decreased wall-time from 209 hours to 71 hours (98.6%).
SHB: 36 cores completed only 81% in 336 hours. Increase to 108, allowed job completion in 160 hours.

Conclusions

- We have demonstrated the efficiency of a Parallel Python Implementation of BLAST+ (PPIB) which leverages HPC resources for putative functional assignments among complex microbiomes which:
  a. is straightforwardly deployable using parallel Python modules
  b. requires minimal HPC system configuration and/or modification
  c. completes BLAST+ functional inference of nearly 250,000 proteins in tractable wall times
  d. is extensible to other large-scale, complex biomolecular datasets

- Given the explosion of biomolecular sequence data in the "-omics" fields, the implementation of PPIB is one solution towards characterizing putative protein functions among complex microbiomes, alleviating a significant bottleneck in data analysis via HPC

Acknowledgements

This work was supported by the Office of Naval Research via NRL core funds. This work utilized HPC resources allocated by the the Department of Defense's High Performance Computing Modernization Program (DoDHPCMP). Further, AJ acknowledges Karles Fellowship Foundation Funds for her fellowship.

The opinions and assertions contained herein are those of the authors and are not to be construed as those of the U.S. Navy, military service at large or U.S. Government.