High Performance Genome Scale Comparisons for the SAGE Method Utilizing Cray Bioinformatics Library Primitives

Eric Stahlberg
The Ohio Supercomputer Center
Cray User Group Meeting
May 20, 2004

OSC

Ohio Supercomuter Center

- **OSC holds a unique niche among Supercomputer Centers**
  - Primarily state funded; allows flexibility
  - Inclusive of all Ohio universities and colleges – not merely a subset of institutions
  - OSC foundation:
    - reliable network, computational services, & training with an impact
    - TFN (OARnet) and HPC form complimentary foundation
- **Excellent reputation, but not as well publicized as other centers in Ohio and out**

OSC

OSC Hardware On Floor

- ~2 TeraFLOP
- 300 GB total memory
- Common home file system
- Command Line and Portal access
- 500 TB Storage Tank

Cray SV1

SGI Altix

Sun Fire 6800

Xeon Cluster

HP Itanium2 Cluster

BALE Cluster

Introductions and Acknowledgements

- Guo-Liang Wang: PI and lead researcher on rice and rice pathogens (OSU)
- Malali Gowda: Post-doc in Wang lab directing the specific project analysis (OSU)
- Shankar Manikantan: Graduate assistant developing critical Java pre/post processing elements (OSC)
- Jeff Doak: Engineer/Analyst doing heavy performance lifting (Cray)
- Eric Stahlberg: Technical lead (OSC)
- Acknowledgements: NSF for rice project, Cray for hardware access

OSC

SAGE Rice Project Description

SAGE Gene Identification in Rice

"Rice is the largest cereal staple for the world's population"

- **SAGE method originally developed at Johns Hopkins for investigating cancer**
- **Method uses characteristic DNA code present in genes as markers to qualitatively and quantitatively measure expression in healthy and diseased plants**
- **Analysis demands comparing up to millions of tags with hundreds of thousands of sequences *per species***

- **Employing OSC Cray SV1 special capabilities and Cray Bioinformatics Library API for custom high-volume SAGE comparisons (already benchmarked on Cray X1)**

The SAGE Method

- SAGE = Serial Analysis of Gene Expression
- Originally developed for cancer research at Johns Hopkins (1995)
- Characteristic signatures identified in DNA and RNA
- Qualitative and Quantitative validity
- Map characteristic signatures to location and function for gene annotation

OSC

The SAGE Tag

ATGAGACAGACGTACGACATGACGTACGTATGGTTAATGGA

- Four nucleotide locus of interest – CATG
- Maps into regions of interest in RNA/DNA
- Extends limited distance for uniqueness
  - Original SAGE 14 nucleotides
  - RL-SAGE extends to 21 nucleotides
- Both direct and reverse complement identification needed

OSC

The SAGE Method for Rice Studies

- Rice is a good genome for study
  - Scientifically: sequenced, good prototype
  - Agriculturally: worldwide dependence, improved productivity has major value
- Computationally
  - Reasonable size: 450 Megabases
- Goals of research
  - Annotate gene function
  - Characterize plant response to rice blast disease

OSC

Problem Definition – Challenge of Terminology

- Scan SAGE tags through chromosomes, EST and cDNA sequences
- Account for potentially sequencing errors
- Track location and sense of match

- Locate substrings in long strings
- Allow for single or double character mismatches
- Record position and whether match was a reverse complement

"The Need for Speed"

- Automatically generating thousands of sequences and tens of thousands of tags
- Analysis time needs to be short to keep from being made obsolete
- New methods generate even more SAGE tags

Why Choose CBL and PCBL?

- Desire to stay on high performance platforms
- Leverage the work of others –
  - primitive methods that are proven
- Easy transition between application development and benchmarking

OSC

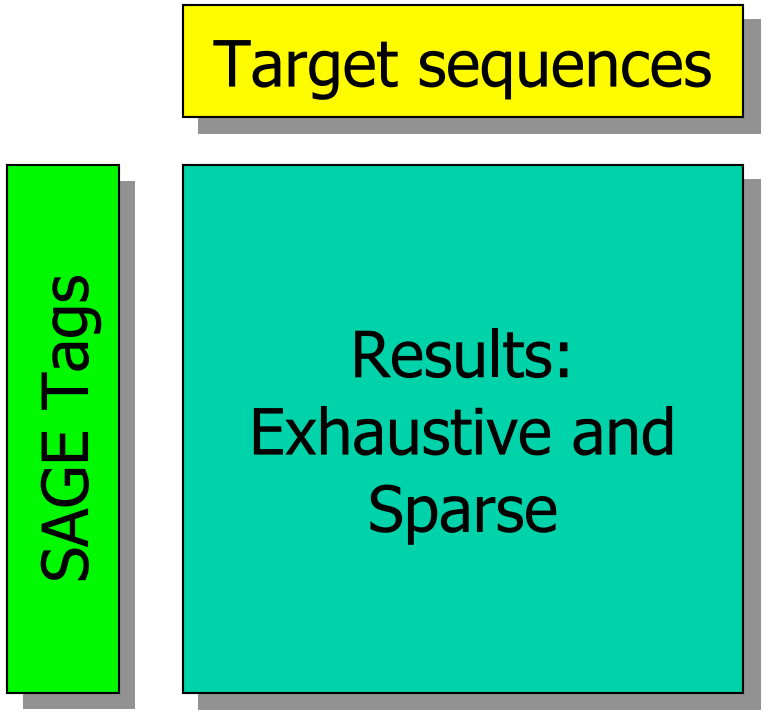
Speed Improvement with cb_read_fasta()

- Memory mapping of entire file makes indexing to elements very fast
- Need to be able to read entire dataset
- Portability – pick the right API

OSC

The Simple Approach First

- Search all tags against all targets
- O(nm) complexity
- Works and efficient in parallel (15.9/16)
- Not yet fast enough
- Lots of misses
- Use cb_searchn()

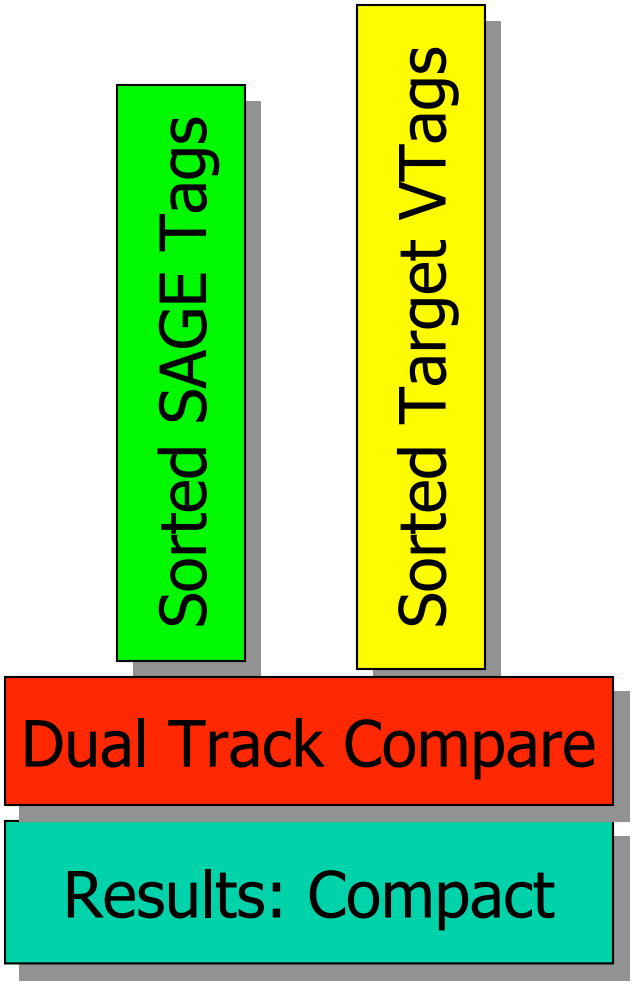


A Reformulation to Exact Match Only

- Exact matches are the most important element to the research project
- Need to screen multiple target lists quickly
- Search for sequencing errors later

OSC

A Refined Approach For Speed



- Sort all tags and candidate tags in targets
- Better than O(n lgn + m lg m) complexity
- Does not use cb_searchn()

The Good, Bad and the Ugly

Good

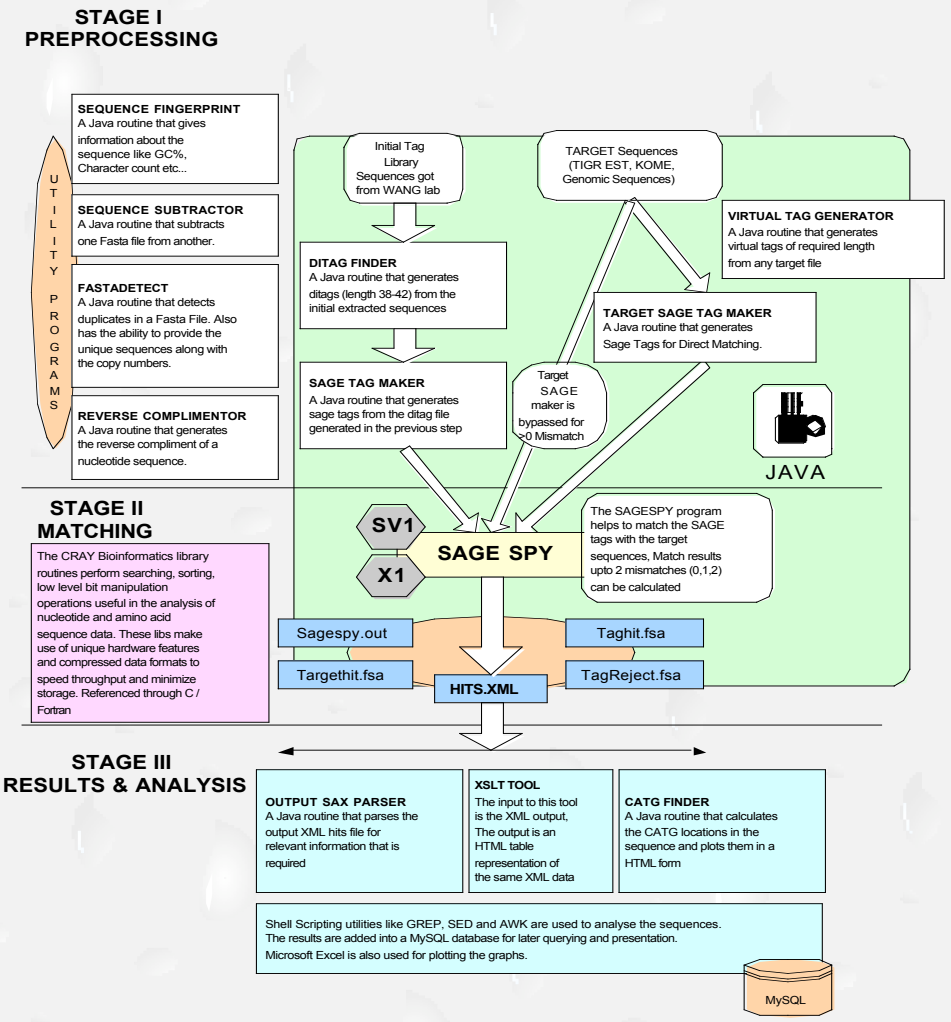
- Can be 4000x faster
- Generalizable to a point

Bad

- Increases number of elements to compare
- Limited to exact matching
- Requires preprocessing

OSC

SAGE ANALYSIS (OSC and Dept. of PLANT PATHOLOGY)



One at a Time vs. Target Fusion

One Target at a Time

```
DO I=1,nqueries,1
   DO j=1,ntargets,1
         CALL cb_searchn()
         Save results
   ENDDO
ENDDO
```

Target Fusion to One Target

```
DO I=1,nqueries,1
         CALL cb_searchn()
         Discard bad results
ENDDO
```

Reduced overhead on calls speeds >30x

OSC

Something Completely Different

- Using XOR for tag matches
- Limitations
  - Target and tags same length (no more than 32 nucleotides)
  - Preprocessing required
- Enhancements
  - No calling overhead
  - Dramatic 500-2000x speedup
- New API

  obl_short_searchn(threshold,len,nq,qd,nt,td,maxhits,hits,nhits)

OSC

Cray SV1 and X1 cb_searchn Routine Performance Details

- SV1 cb_searchn
  - version is written in CAL
  - 6 functional units for bit operations
  - BMM is full speed
  - has snake shift
- X1 cb_searchn
  - version is Fortran version of SV1 CAL code
  - 3 functional units for bit operations
  - BMM is half speed
  - does not have snake shift
- End result: X1 is 1.3x SV1 performance

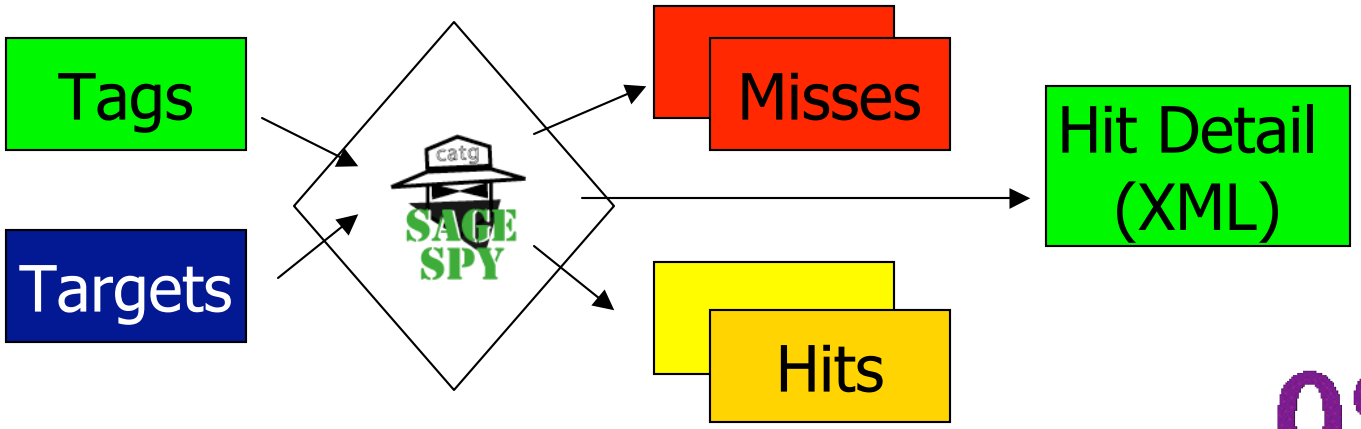
OSC

Comparative Timings X1 and SV1 (100 Tag Benchmark Set)

100 Tag Set	Number of Targets	SV1 XOR	X1 XOR	SV1 Fused	X1 Fused	SV1 single	X1 single
Chrom1	634952	1.35	0.36	33.57	27.80	716.42	1372.61
Chrom2	566094	1.20	0.32	30.22	25.11	638.72	1235.54
Chrom3	602420	1.29	0.34	32.93	27.40	679.71	1301.10
Chrom4	472536	1.00	0.27	25.82	21.54	533.17	1026.72
Chrom5	410074	0.87	0.24	21.98	18.11	462.69	890.75
Chrom6	449616	0.96	0.26	23.90	19.78	507.30	973.52
Chrom7	417020	0.89	0.25	22.24	18.39	470.52	902.95
Chrom8	403882	0.86	0.23	21.50	17.94	455.70	871.40
Chrom9	331252	0.71	0.19	17.71	15.18	373.75	714.06
Chrom10	284020	0.61	0.16	15.77	13.19	320.46.	618.16
Chrom11	364548	0.77	0.20	20.20	16.89	411.32	792.54
Chrom12	360950	0.77	0.21	20.01	16.73	407.26	782.21
TIGR	470924	1.00	0.26	40.29	33.87	531.35	1015.72
KOME	475998	1.01	0.27	32.17	27.11	537.07	1026.53

SAGESPY Core Application

- Uses tag and target files as input
- Creates FASTA files of tags and targets matched
- Creates FASTA files of tags and targets not matched
- XML detail file of match characteristics
- Latest versions proven on Cray hardware



SAGESPY Availability

- General availability later this summer
- Adopting portable I/O APIs
- Incorporating maximum speed improvements
- Quality assurance prior to distribution
- Java processing suite
- Available via Ohio Bioscience Library (OBL)
  - Relies on CBL and PCBL
  - Extensions to CBL and PCBL APIs
  - Applications and higher level abstractions

OSC

Conclusions

- CBL and PCBL are viable APIs for development
- Higher level APIs are useful for application development
- Portability and implementation compatibility a detail to address
- Optimization has lead to improvements in time to solution greatly exceeding parallel improvements alone

OSC