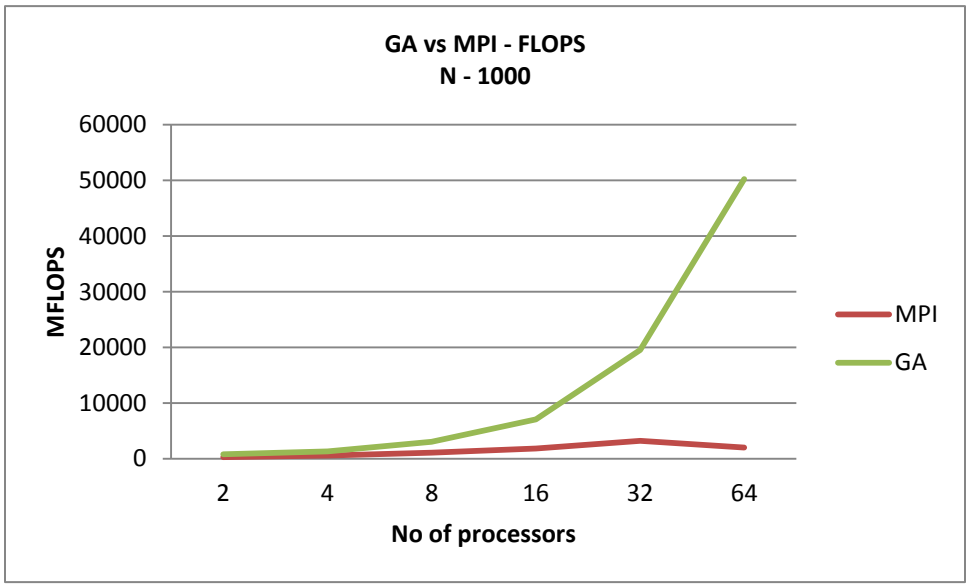
Comparative performance study of MPI and Global Arrays

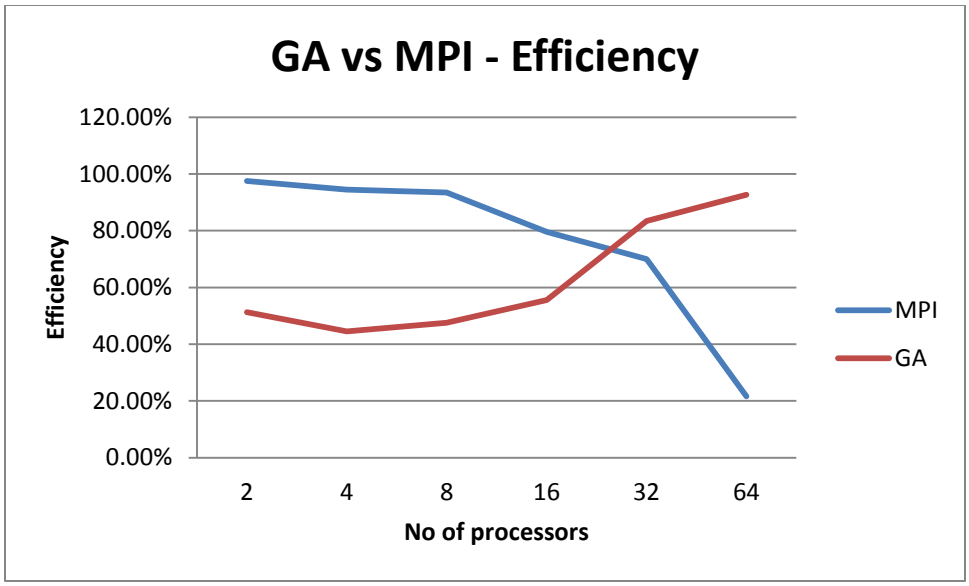
Benchmark: Matrix multiplication of two 1000*1000 square matrices

No of processors	MFLOPS		Efficiency	
	MPI	GA	MPI	GA
2	264.2	776.3	97.5%	51.3%
4	534.76	1308.7	94.5%	44.5%
8	1054.85	3033.3	93.4%	47.6%
16	1813.24	7049.6	79.6%	55.6%
32	3164.56	19496.0	70.0%	83.4%
64	1960.78	50204.7	21.6%	92.7%

Serial runtime: Runtime on one processor

MPI – 14.75 s GA – 2.67 s





The Efficiency plot indicates that GA may be more scalable as the size of the problem increases whereas MPI is suitable for relatively smaller problem size.

The amount of communication required by the program is $3*N^2$. i.e.

N^2 - for broadcasting array B

N^2 - for sending out each interval of rows ($P*(N^2/P) = N^2$)

N^2 - for receiving the results back from the worker processors.

The number of calculations required by the algorithm is N^3.

Communication – computation ratio = $T_{COMM}*3*N^2 / T_{CALC}*N^3$.

As N approaches infinity, we get ZERO. This implies that the communication overhead reduces significantly with increase in problem size.

Let's the ratio T_{COMM}/T_{CALC} is 1. Then, taking into account the cost of communication, the number of operations (both calculations and communications) performed by each of the P processors is

$N^3/P + N^2 + 2N^2/P$

Where, N^3/P – computation cost, $N^2 + 2N^2/P$ - communication cost

The cost of communication is the cost of receiving the broadcast plus the costs of receiving and sending the rows it's responsible for. If we compare this value with the conventional cost of computation in the case of a single processor (N^3), we get:

$$(3N^3 + (P+2)*N^2) / \quad P*N^3$$

The limit as N approaches infinity is 1/P i.e. if the overhead of communication can be factored out, then as the size of the matrix increases then using P processors will speed up the problem by P times.

Let us consider the addition of two matrices. This algorithm is only N^2. The ratio of communication over computation on P processors is

$$3N^2 / N^2 = 3$$

This implies that the parallel implementation of the algorithm is efficient if and only if T_{COMM}/T_{CALC} is less than 1/3.

Plan for remainder of work:

Collect profile information from a few more benchmarks and also compare the efficiency of the implementations of MPI and GA primitives, so as to identify the bottlenecks and the underlying tradeoffs.