

Optimization technology is based on applied mathematics and computer science, and is widely used to help business people make better decisions. It can quickly determine how to most effectively allocate resources, automatically balance trade-offs and business constraints. It eliminates the need to manually work out ideas, plans and schedules, so you get maximum operational and business efficiency. Star-P provides a platform for large scale optimizations to be performed on high performance parallel systems.

Key benefits

- **Maximum operational efficiency**
Improve utilization for any sort of resource: capital, personnel, equipment, vehicles, facilities. Assign the best resources to each task, at the best possible time.
- **Uncover solutions to the toughest challenges**
Explore alternatives in minutes. Cope with the most difficult trade-offs that nobody had ever considered. Extract the maximum yield from every resource. Cope with the toughest conflicts.
- **Measurable return on investment, fast**
See results within months, or even weeks. Costs drop, earnings increase and service improves. Customers are happier, and so are your employees. Some applications save thousands of dollars a year, and others save millions—but there is always a return on investment (ROI).
- **Applications in every industry**
Optimization is at work everywhere: manufacturing, transportation, logistics, financial services, utilities, energy, telecommunications, government, defense and retail.

Why Interactive Supercomputing & Star-P in Optimization?

ISC is a leader in the field of parallel programming using very high level languages and bridging with high performance technical computing. In the discipline of optimization, we have incorporated some of the world's most advanced optimization libraries for solving tough business and research problems. These libraries solve constrained and unconstrained continuous and discrete problems. The availability of linear programming, quadratic programming, solving systems of nonlinear equations, multi-objective optimization, nonlinear optimization, nonlinear least squares, and binary integer programming allow for solutions on large and complex problems. In addition, ISC has partnered with companies that deliver an even larger suite of libraries and solvers to deliver optimization solutions for parallel computing systems architectures.

Universities use Star-P for research and teaching while Government agencies, Financial Institutions, and numerous other commercial enterprises use Star-P to solve their largest and most complex optimization problems.

Example Applications Using Star-P in Optimization

Linear and Quadratic Programming

Optimization techniques such as linear and quadratic programming are widely used in a broad range of research and analysis, primarily in the financial and engineering fields. These optimization tools have been proven useful in modeling diverse types of problems in planning, routing, scheduling.

Star-P enables easy and effective scaling of optimization applications through parallel computing. Moreover, Star-P enables further order-of-magnitude performance improvement of optimization algorithms by providing means to easily include specialized optimization libraries such as CLP or CPLEX.

Dramatic improvement of performance achievable easily with Star-P through the combination of parallel computing and inclusion of a specialized optimization library is illustrated in the examples shown below.

Linear Programming Example

The following code represents an example of solving a batch of linear programming optimization problems in a serial MATLAB® session:

```
% min f'*x      subject to:   A*x <= b
%           x                x >= lb
%
n = 200;
m = n;
numiters = 16;
f = rand(n,1);
A = -rand(m,n);
b = -rand(m,numiters);
lb = zeros(n,1);           %Lower Bound at zero

x = zeros(n,numiters);
fval = zeros(1,numiters);
tic
for k = 1:numiters
    [x(:,k),fval(k)] = linprog(f,A,b(:,k),[],[],lb);
end
desktop_time = toc;
```

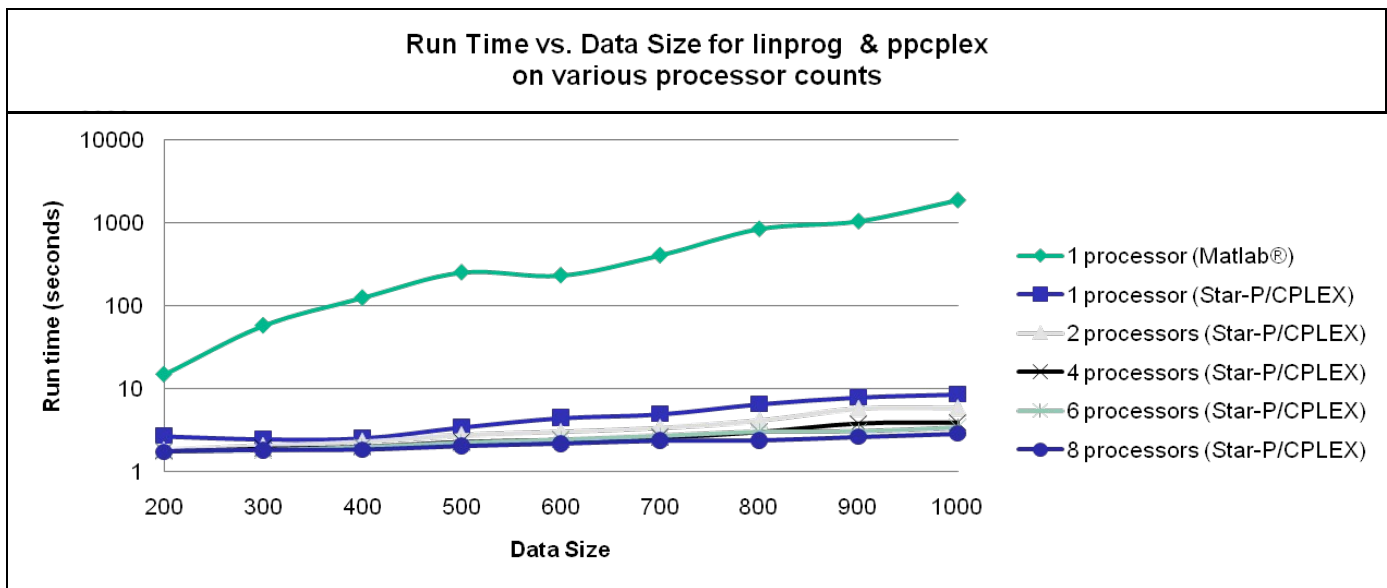
The same set of problems can be solved using Star-P and its `ppcplex` interface, which

- Provides direct task parallel interface for executing functions from the CPLEX optimization library
- Executes an individual CPLEX session on each separate instance of Star-P's Task Parallel Engine
- Provides appropriate argument checking when splitting or broadcasting separate input arguments

```
f_p = ppback(f);
A_p = ppback(A);
b_p = ppback(b);
lb_p = ppback(lb);

% linprog and loop equivalent using Star-P's Task Parallel CPLEX interface
tic
[x_p,fval_p] = ppcplex([],bcast(f_p),bcast(A_p),bcast(b_p),[],[],[],[],bcast(lb_p));
starp_time = toc;
```

The runtimes measured for a desktop running this `linprog` code for various data sizes in MATLAB® are compared below with the corresponding runtimes achieved with Star-P on a server using from 1 to 8 processors (note the log scale).



Quadratic programming example

The following code shows an example of solving a batch of quadratic programming optimization problems in a serial MATLAB® session:

```
n = 200;
m = n;
numiters = 16;
H = rand(n);
H = H'*H; % A symmetric positive definite quadratic
constraint matrix
f = randn(n,1);
A = randn(m,n);
b = randn(m,1);
x0 = repmat([(1:n)', rand(n,1)], [1,1,n]); % Initial Guesses for the optimal solution
of each separate optimization.

x = zeros(n,numiters);
fval = zeros(1,numiters);
tic
for k = 1:numiters
    [x(:,k), fval(k)] = quadprog(H,f,A,b,[],[],[],[],x0(:,2,k));
end
desktop_time = toc;
```

The same set of problems can be solved using Star-P and its ppcplex interface, as shown below:

```
H_p = ppback(H);
f_p = ppback(f);
```

```

A_p = ppback(A);
b_p = ppback(b);
x0_p = ppback(x0);

% quadprog and loop equivalent using Star-P's Task Parallel CPLEX interface
tic
[x_p,fval_p] = ppcplex(bcast(H_p),bcast(f_p),bcast(A_p),bcast(b_p),[],[],[],[],[],[],[],
[],[],[],[],[],[],[],[],[],x0_p);
starp_time = toc;

```

The runtimes measured for a desktop running this quadprog code for various data sizes in MATLAB® are compared below with the corresponding runtimes achieved with Star-P on a server using from 1 to 8 processors (note the log scale).

