CMPS 373: *Parallel Computing*		June 7, 2005
	Final Exam	
Instructor: Fatima Abu Salem	**Name:**	

Duration: 90 minutes

Question 1 (60pts)

Let A be a sparse $m \times m$ matrix and B a dense $m \times n$ matrix with $m \geq n$. Consider the matrix-matrix multiplication $C = AB$.

(a) What is the time complexity of a straightforward sequential algorithm?

(b) Choose distributions for A, B, and C and formualte a corresponding parallel algorithm. **Motivate your choices of any distribution you adopt.**

(c) Analyse the time complexity of the parallel algorithm.

Question 2 (40pts)

The Conjugent Gradient (CG) method is an iterative method for solving a symmetric positive definite linear system of equations $A\mathbf{x} = \mathbf{b}$. (A matrix A is **positive definite** if $\mathbf{x}^{\mathrm{T}} A\mathbf{x} > 0$ for all $\mathbf{x} \neq \mathbf{0}$.)

The algorithm computes a sequence of approximations $\mathbf{x}^k$, $k = 0, 1, 2, \ldots$, that converges towards the solution $\mathbf{x}$. The algorithm is usually considered converged when $||\mathbf{r}^k|| \leq \epsilon_{conv}||\mathbf{b}||$, where $\mathbf{r}^k = \mathbf{b} - A\mathbf{x}^k$. Recall that for a vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, the (Euclidean) norm is given by $(x_1^2 + x_2^2 + \ldots + x_n^2)^{1/2}$.

A sequential CG algorithm is given on the next page [1].

(a) Design a parallel CG algorithm based on the sparse matrix-vector multiplication discussed in class. How do you distribute the various vectors and matrix A? **Motivate your design choices.**

(b) Analyse the time complexity of your algorithm.

[1] All bold-face variables denote vectors, and non-bold ones denote scalars

```
input: A: sparse n × n matrix,
       b: dense vector of length n.
output: x: dense vector of length n, such that Ax ≈ b.

x := x^0; { initial guess }
k := 0; { iteration number }
r := b − Ax;
ρ := ||r||^2;
while √ρ > ε_conv||b||&&k < k_max do
        if k = 0 then
              p := r;
        else
              β := ρ/ρ_old;
               p := r + βp;
        w := Ap;
        γ := p^T w;
        α := ρ/γ;
        x := x + αp;
        r := r − αw;
        ρ_old := ρ;
        ρ := ||r||^2;
        k := k + 1;
```