PDC Lab: Scheduling

Shortest Path algorithm using openmp:

Name:Raghu.N
Adno:16bce1260

Here we use the various scheduling techniques and check the running time for each kind.The algorithm we use here is Djkstras shortest part algorithm.

Code:

```
#include <stdio.h>
#include <time.h>
#include <math.h>
#include <omp.h>


#define INT_MAX 100000
#define TRUE    1
#define FALSE   0
#define V 8
#define E 11

//boolean type
typedef int bool;

//Represents an edge or path between Vertices
typedef struct
{
	int u;
	int v;

} Edge;

//Represents a Vertex
typedef struct
{
	int title;
	bool visited;

} Vertex;

//Prints the array
void printArray(int *array)
{
	int i;
	for(i = 0; i < V; i++)
	{
		printf("Path to Vertex %d is %d\n", i, array[i]);
	}
```

```
}

//OpenMP Implementation of Dijkstra's Algorithm
void DijkstraOMP(Vertex *vertices, Edge *edges, int *weights, Vertex *root)
{


	double start, end;





	root->visited = TRUE;

	int len[V];
	len[(int)root->title] = 0;

	int i, j;

	for(i = 0; i < V;i++)
	{

		if(vertices[i].title != root->title)
		{
			len[(int)vertices[i].title] = findEdge(*root, vertices[i], edges, weights);


		}
		else{

			vertices[i].visited = TRUE;
		}


	}


	start = omp_get_wtime();
	for(j = 0; j < V; j++){


		Vertex u;
		int h = minPath(vertices, len);
		u = vertices[h];


		//OpenMP Parallelization Starts here and we are using dynamic scheduling!!!
		#pragma omp parallel for schedule(dynamic) private(i)
			for(i = 0; i < V; i++)
			{
```

```
                        if(vertices[i].visited == FALSE)
                        {
                                int c = findEdge( u, vertices[i], edges, weights);
                                len[vertices[i].title] = minimum(len[vertices[i].title],
len[u.title] + c);

                        }
                }



        }
        end = omp_get_wtime();
        printArray(len);
        printf("Running time: %f ms\n", (end - start)*1000);



}
```

Output:

Dynamic:

```
raghu@raghu-Lenovo-G50-80: ~/Desktop
raghu@raghu-Lenovo-G50-80:~/Desktop$ ./sched
Minimum Distances of the vertices:
Path to Vertex 0 is 0
Path to Vertex 1 is 40
Path to Vertex 2 is 30
Path to Vertex 3 is 40
Path to Vertex 4 is 10
Path to Vertex 5 is 40
Path to Vertex 6 is 30
Path to Vertex 7 is 30
Running time: 2.277920 ms
raghu@raghu-Lenovo-G50-80:~/Desktop$
```

Static:

```
raghu@raghu-Lenovo-G50-80: ~/Desktop
raghu@raghu-Lenovo-G50-80:~/Desktop$ ./sched
Minimum Distances of the vertices:
Path to Vertex 0 is 0
Path to Vertex 1 is 40
Path to Vertex 2 is 30
Path to Vertex 3 is 40
Path to Vertex 4 is 10
Path to Vertex 5 is 40
Path to Vertex 6 is 30
Path to Vertex 7 is 30
Running time: 6.954352 ms
raghu@raghu-Lenovo-G50-80:~/Desktop$
```

Guided:

```
raghu@raghu-Lenovo-G50-80: ~/Desktop
raghu@raghu-Lenovo-G50-80:~/Desktop$ ./sched
Minimum Distances of the vertices:
Path to Vertex 0 is 0
Path to Vertex 1 is 40
Path to Vertex 2 is 30
Path to Vertex 3 is 40
Path to Vertex 4 is 10
Path to Vertex 5 is 40
Path to Vertex 6 is 30
Path to Vertex 7 is 30
Running time: 2.758995 ms
raghu@raghu-Lenovo-G50-80:~/Desktop$
```