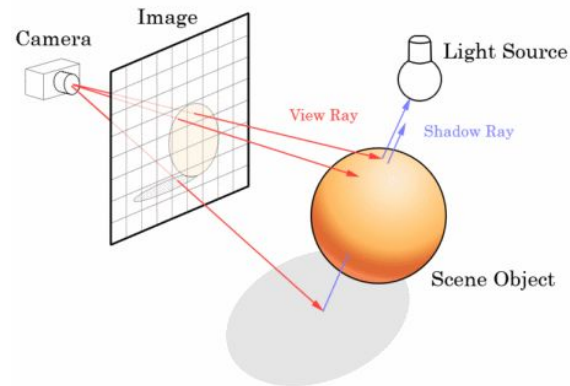


Ray Tracing

Robert Hughes

What is Ray Tracing

- Computer Graphic Rendering Technique
- Traces the path of light as pixels in an image plane
 - Simulates the effects of the rays on objects
- Has a high level of realism
 - Simulates variety of optical effects (reflection, refraction, scattering etc.)
- High computational cost
- Good For ahead of time rendering
 - Movies
 - Television
 - Special Effects
- Bad For real time rendering
 - Video games



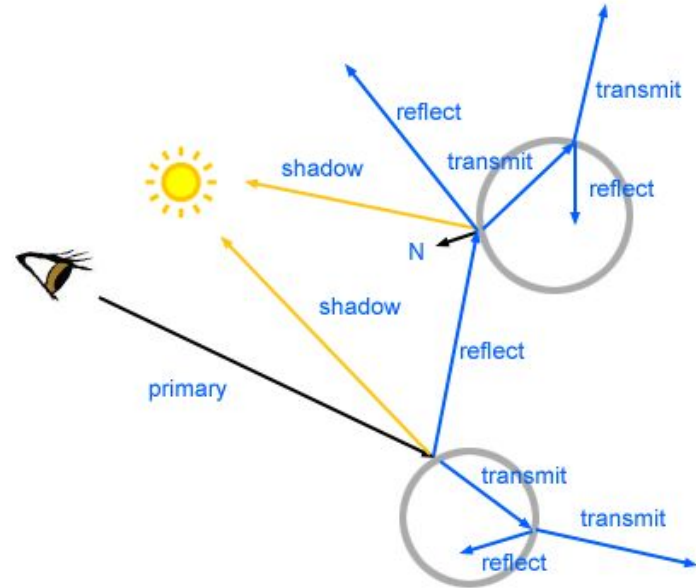
Design of Parallel Solution

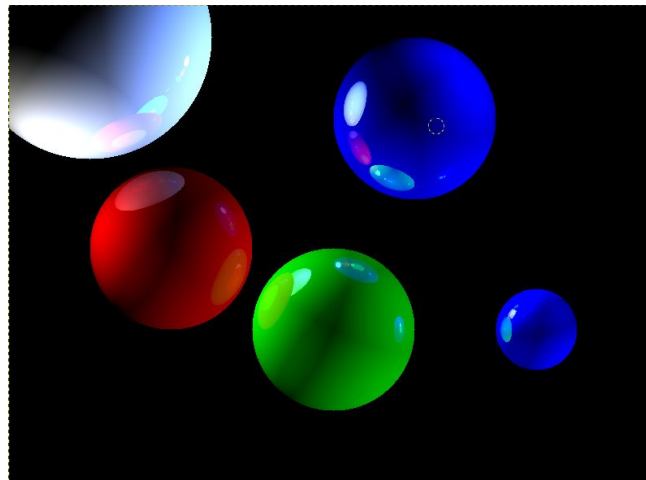
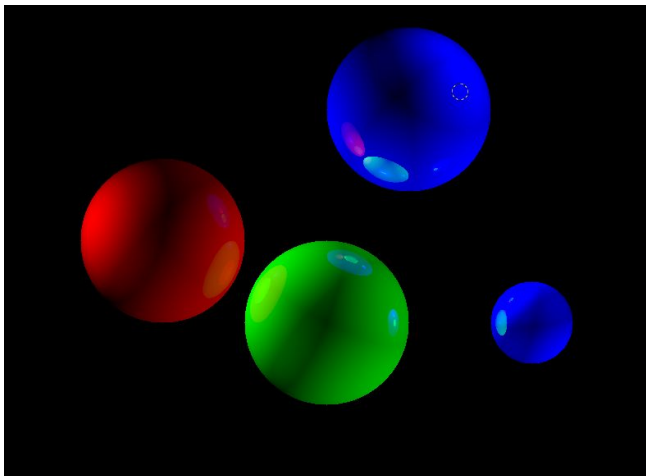
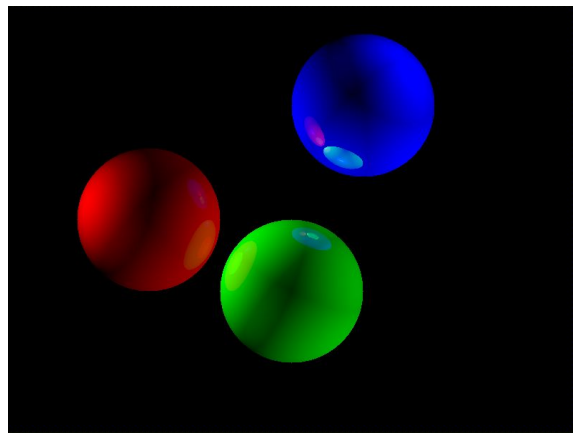
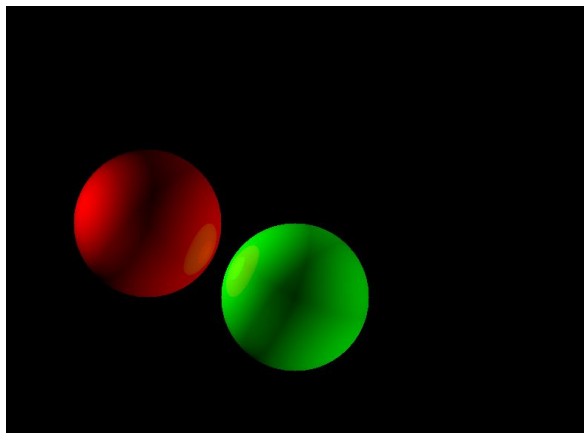
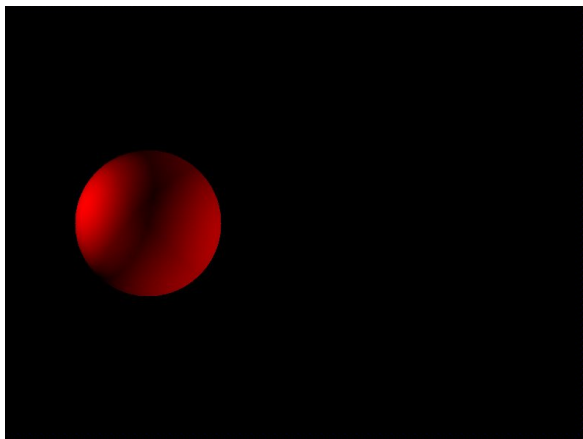
- Pthreads
- Raw image stored in a single array to be written to a file
 - `unsigned char img[3*WIDTH*HEIGHT];`
- Each thread accesses a part of the array to write the data to the array
 - `for (y=my_start; y<my_end; y++) {`
 - `for (x=0; x<WIDTH; x++) {`
- No need for critical sections

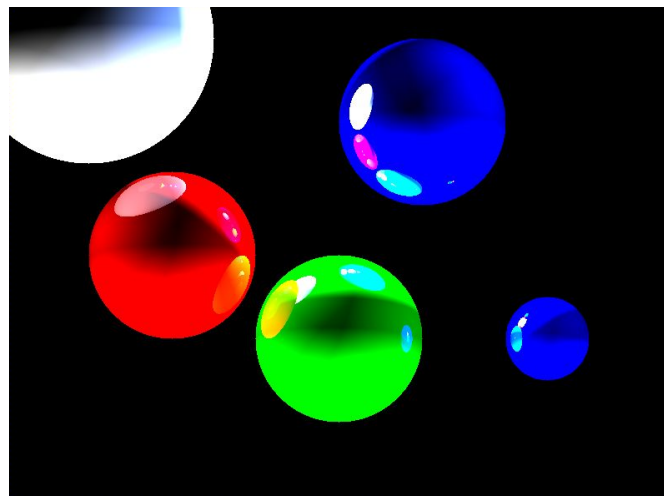
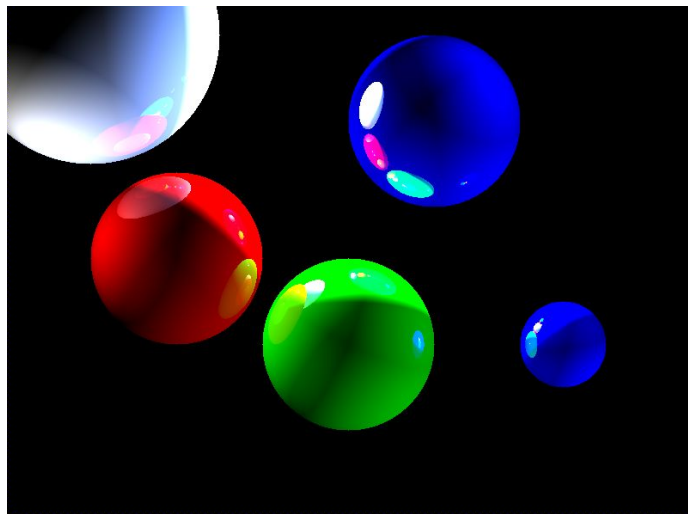
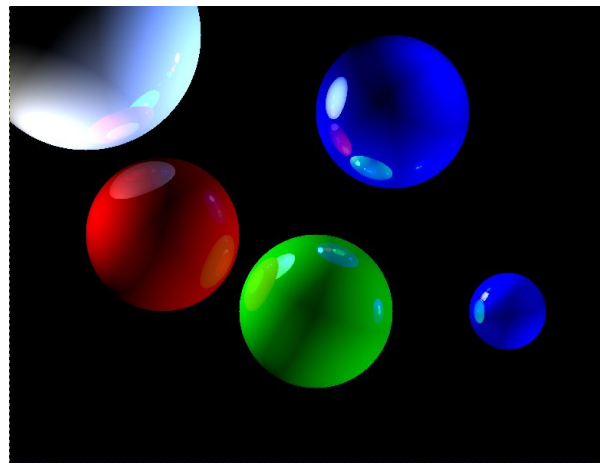
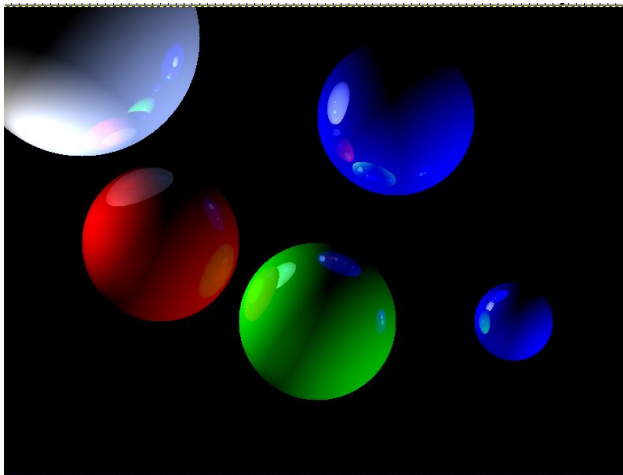
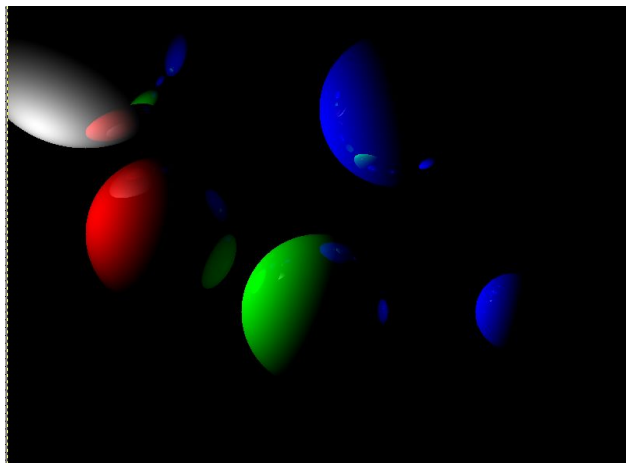
```
207 int x, y;
208 for(y=my_start; y<my_end; y++){
209     for(x=0; x<WIDTH; x++){
210
211         float red = 0;
212         float green = 0;
213         float blue = 0;
214
215         int level = 0;
216         float coef = 1.0f;
217
218         r.start.x = x;
219         r.start.y = y;
220         r.start.z = -2000;
221
222         r.dir.x = 0;
223         r.dir.y = 0;
224         r.dir.z = 1;
225
226         do{
227             /* Find closest intersection */
228             float t = 20000.0f;
229             int currentSphere = -1;
230
231             unsigned int i;
232             for(i = 0; i < 5; i++){
233                 if(IntersectRaySphere(&r, &spheres[i], &t))
234                     currentSphere = i;
235             }
236             if(currentSphere == -1) break;
237
238             vector scaled = vectorScale(t, &r.dir);
239             vector newStart = vectorAdd(&r.start, &scaled);
240
241             /* Find the normal for this new vector at the point of intersection */
242             vector n = vectorSub(&newStart, &spheres[currentSphere].pos);
243             float temp = vectorDot(&n, &n);
244             if(temp == 0) break;
245
246             temp = 1.0f / sqrtf(temp);
247             n = vectorScale(temp, &n);
248
249             /* Find the material to determine the colour */
250             material currentMat = materials[spheres[currentSphere].material];
251
252             /* Find the value of the light at this point */
253             unsigned int j;
254             for(j=0; j < 4; j++){
255                 light currentLight = lights[j];
```

Experiments

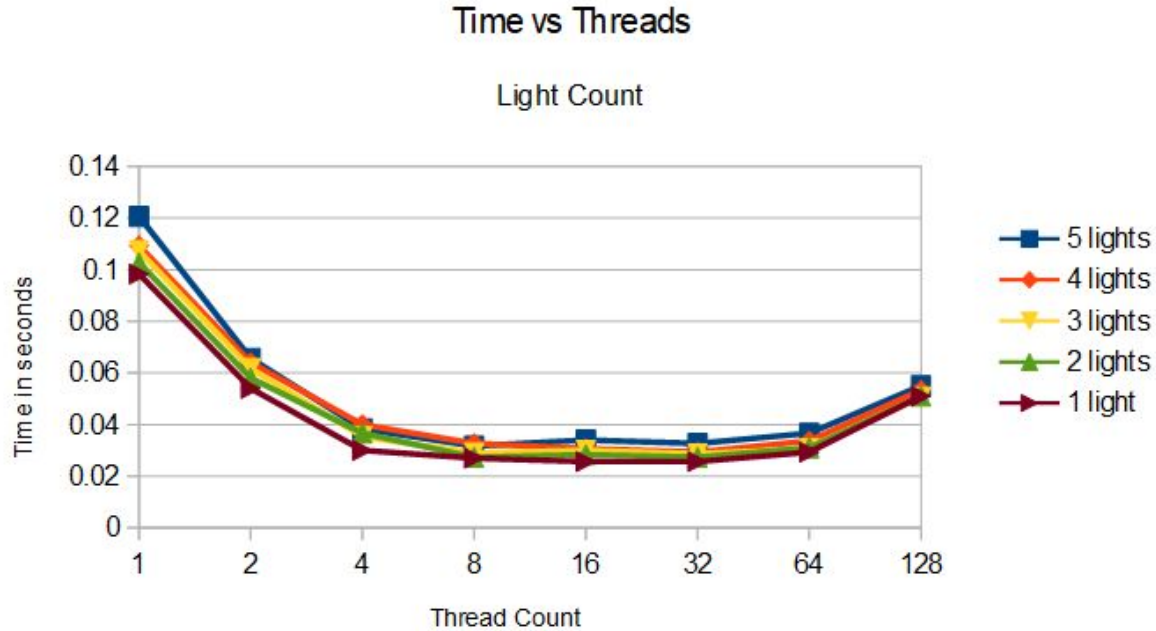
- Collect the timing data for different variations of the program to run with thread counts
 - Different Ball Counts
 - Different Lighting sources



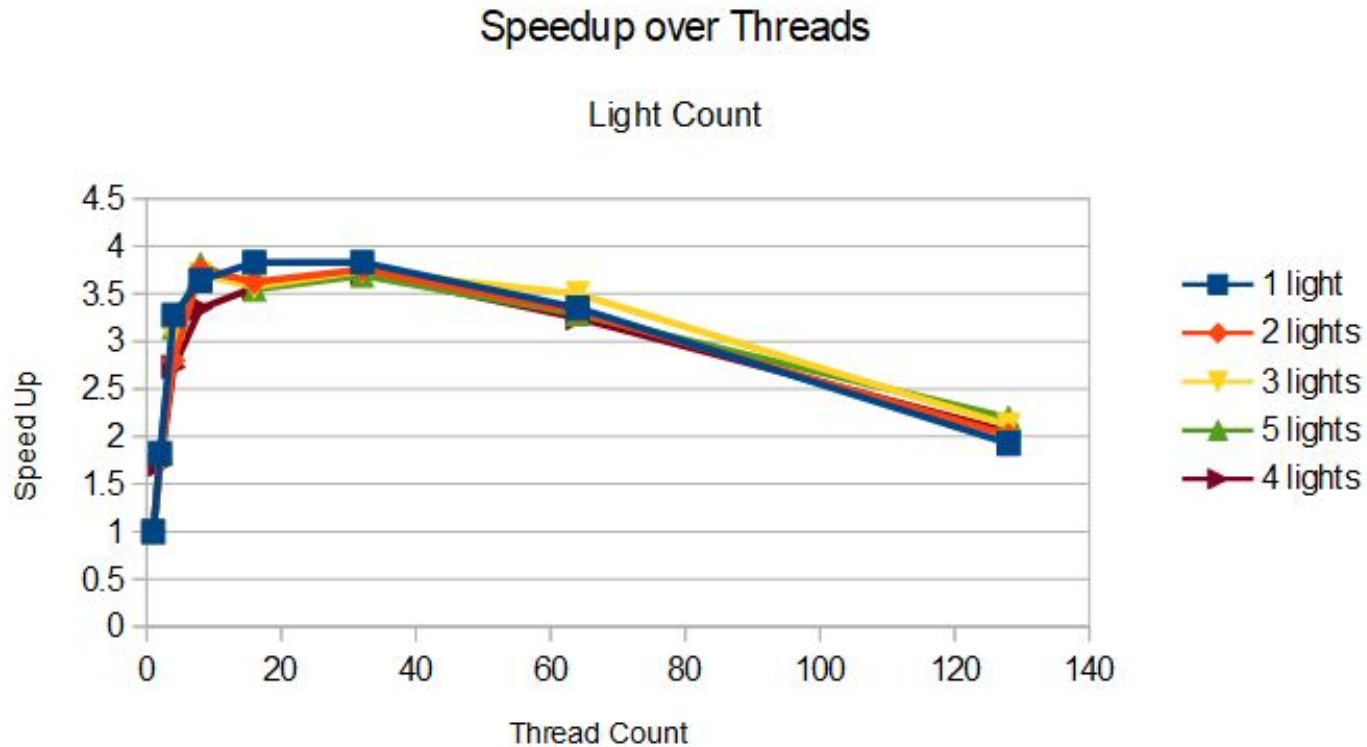




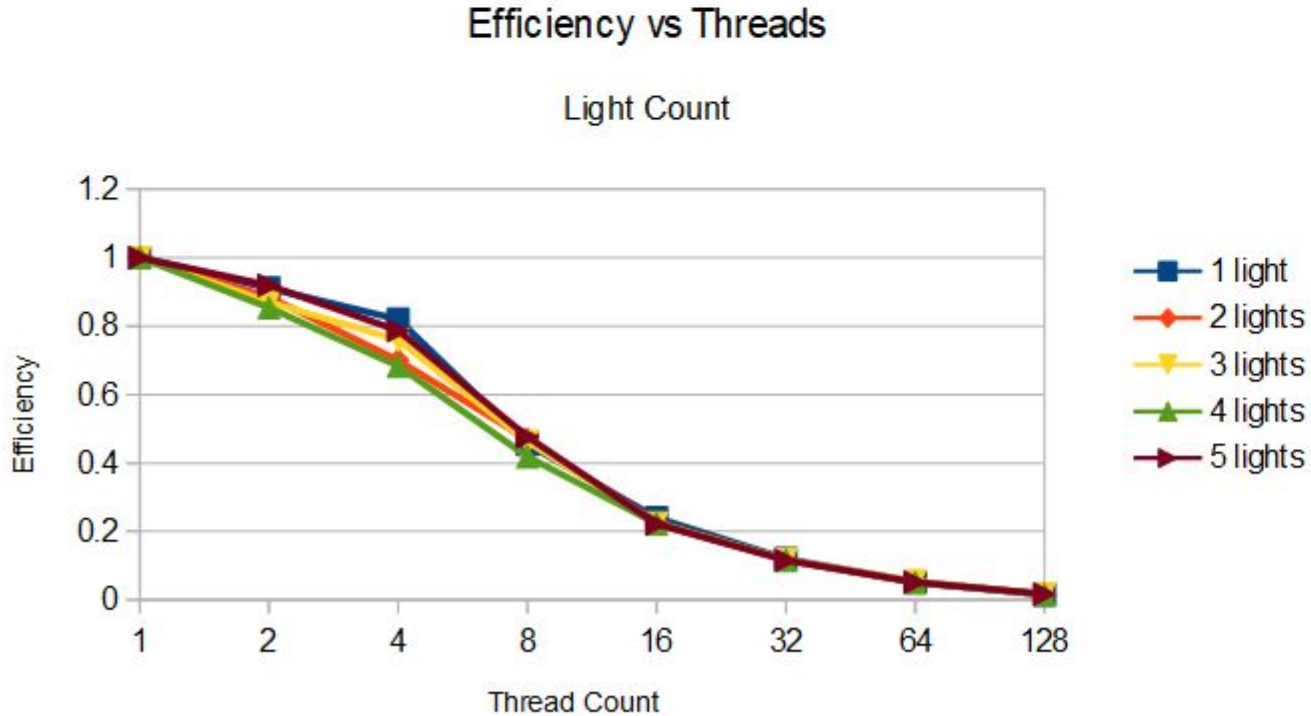
Analysis of Results



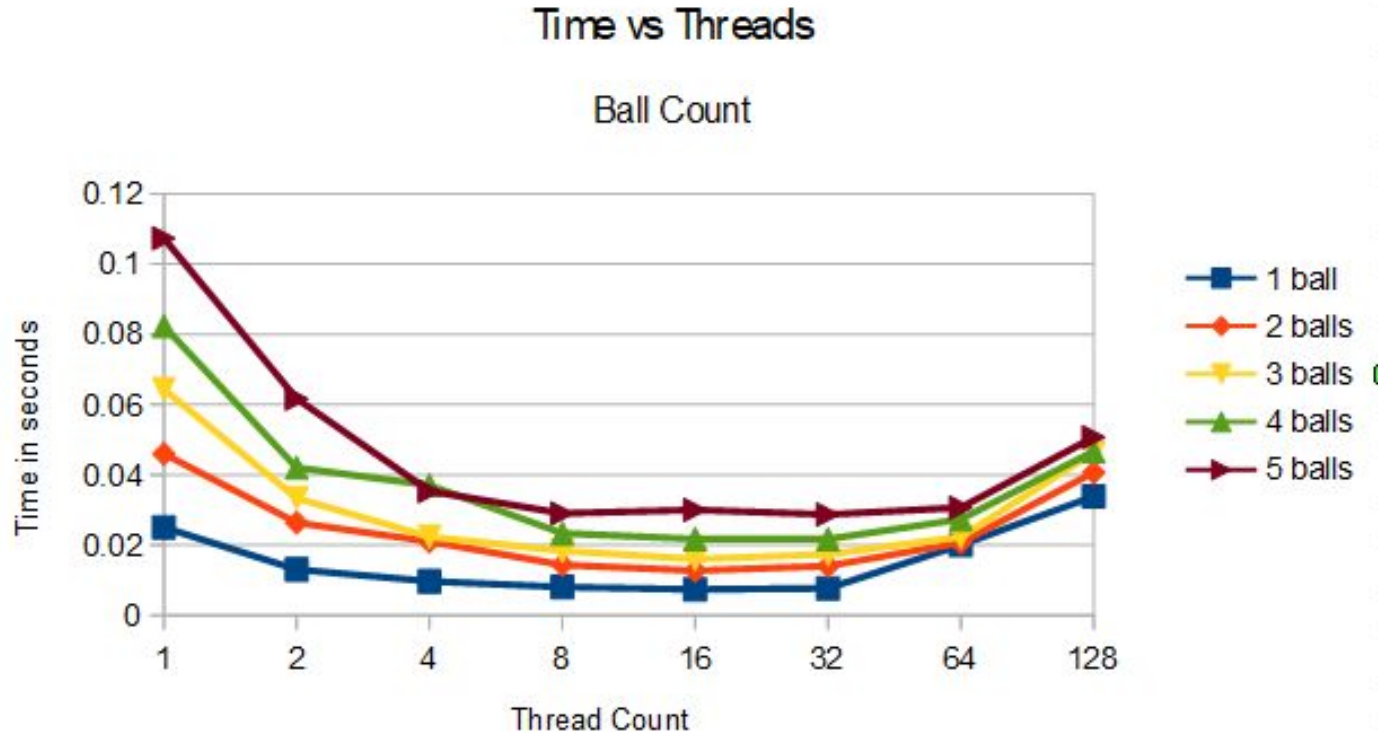
Analysis of Results



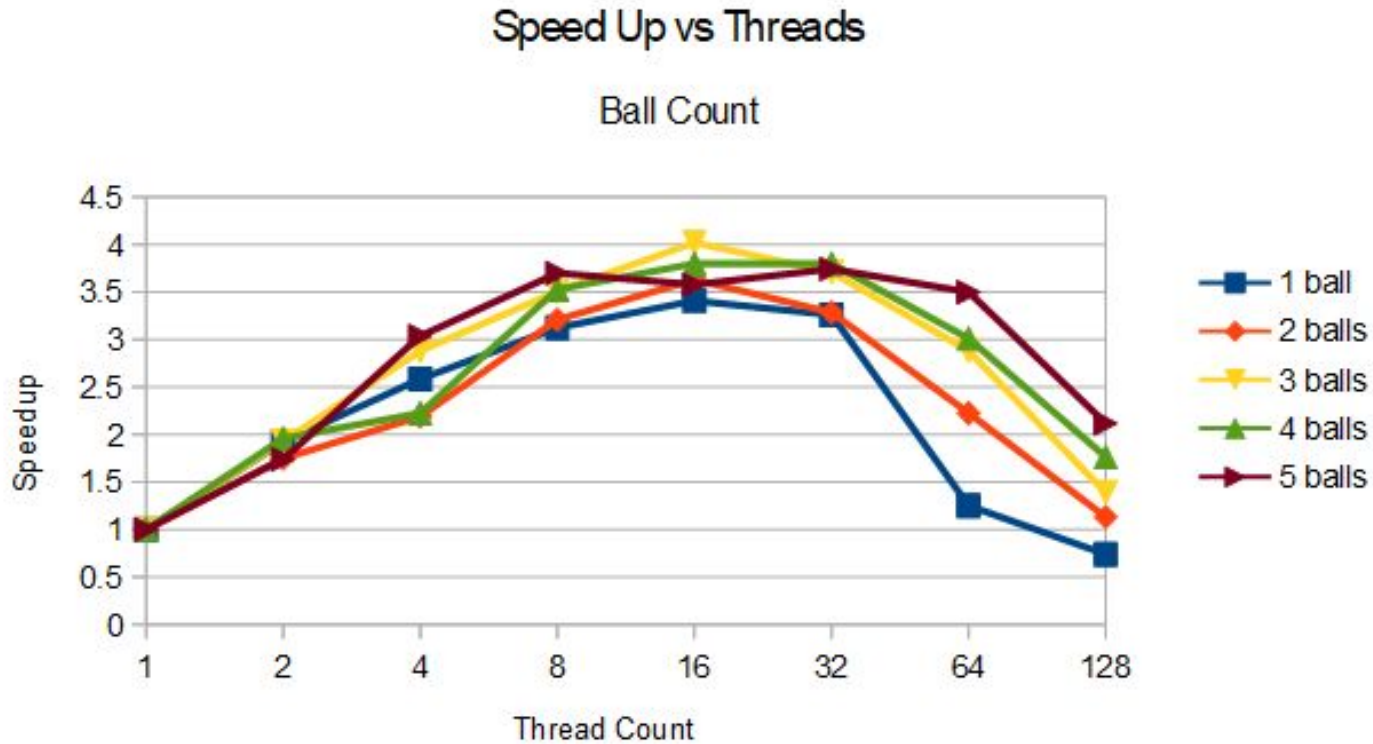
Analysis of Results



Analysis of Results



Analysis of Results



Analysis of Results

