| # | Loop Body | AutoParLLM-GPT-4 suggestion | Developer’s version |
|---|---|---|---|
| 1 | for( int i = 0; i < 12; i++ ) { for( int j = 0; j < num_nucs[i]; j++ ) concs[i * max_num_nucs + j] = LCG_random_double(&seed); } | *NP* | *NP* |
| 2 | for( i = 0; i < in.lookups; i++ ){ .... for(int j = 0; j < 5; j++ ) { ... } verification += max_idx+1; } | *reduction(+:verification)* | *schedule(dynamic, 100) reduction(+:verification)* |
| 3 | for( int m = 0; m < num_mats; m++ ) { if( num_nucs[m] > *max_num_nucs ) *max_num_nucs = num_nucs[m]; } | *NP* | *NP* |
| 4 | for( int j = 0; j < num_nucs[mat]; j++ ){ double xs_vector[5]; p_nuc = mats[mat*max_num_nucs + j]; conc = concs[mat*max_num_nucs + j]; ... } | *NP* | *NP* |
| 5 | for( int i = 0; i < 12; i++ ){ double running = 0; for( int j = i; j > 0; j-- ) running += dist[j]; if( roll < running ) return i; } | *NP* | *NP* |
| 6 | for( i = 0; i < in.lookups; i++ ) { uint64_t seed = STARTING_SEED; ... SD.p_energy_samples[i] = p_energy; SD.mat_samples[i] = mat; } | *private(i)* | *schedule(dynamic, 100)* |
| 7 | for( i = offset; i < offset + num_samples_per_mat[m]; i++) { .... for(int j = 0; j < 5; j++ ) { if( macro_xs_vector[j] > max ) { ... } } verification += max_idx+1; } | *reduction(+: verification)* | *schedule(dynamic, 100) reduction(+:verification)* |
| 8 | for( int l = 0; l < in.lookups; l++ ) { num_samples_per_mat[SD.mat_samples[l]]++; } | *NP* | *NP* |
| 9 | for( int m = 1; m < 12; m++ ) { offsets[m] = offsets[m-1] + num_samples_per_mat[m-1]; } | *NP* | *NP* |
| 10 | for( int m = 0; m < 12; m++ ) { quickSort_parallel_d_i(SD.p_energy_samples + offsets[m], SD.mat_samples + offsets[m], num_samples_per_mat[m], in.nthreads); } | *NP* | *NP* |