|  |
| --- |
| /\* |
| A sample code for N-ary search with OpenMP |
|  |
| To build and execute: |
| $ g++ -fopenmp -O3 nary-search.cpp && ./a.out |
|  |
| \*/ |
|  |
|  |
| #include <iostream> |
| #include <vector> |
| #include <boost/timer.hpp> |
|  |
| enum locate\_t { |
| EQUAL, |
| LEFT, |
| RIGHT |
| }; |
|  |
| template <typename T> int |
| nary\_search(const std::vector<T>& a, T const key, int const N) { |
| std::vector<int> mid(N + 1); |
| std::vector<locate\_t> locate(N + 2); |
|  |
| locate[0] = RIGHT; |
| locate[N + 1] = LEFT; |
|  |
| int lo = 0; |
| int hi = a.size() - 1; |
| int pos = -1; |
|  |
|  |
| double step, offset; |
|  |
| #pragma omp parallel |
| { |
| while(lo <= hi && pos == -1) { |
| #pragma omp single |
| { |
| mid[0] = lo - 1; |
| step = (hi - lo + 1) / (N + 1); |
| } |
|  |
| #pragma omp for private(offset) firstprivate(step) |
| for(int i = 1; i <= N; i++) { |
| offset = step \* i + (i - 1); |
| int const lmid = mid[i] = lo + static\_cast<int>(offset); |
|  |
| sleep(1); |
| if(lmid <= hi) { |
| if(a[lmid] > key) { |
| locate[i] = LEFT; |
| } |
| else if(a[lmid] < key) { |
| locate[i] = RIGHT; |
| } |
| else { |
| locate[i] = EQUAL; |
| pos = lmid; |
| } |
| } |
| else { |
| mid[i] = hi + 1; |
| locate[i] = LEFT; |
| } |
| } |
| #pragma omp single |
| { |
| for(int i = 1; i <= N; i++) { |
| if(locate[i] != locate[i-1]) { |
| lo = mid[i - 1] + 1; |
| hi = mid[i] - 1; |
| } |
| } |
| if(locate[N] != locate[N+1]) { |
| lo = mid[N] + 1; |
| } |
| } // end of single |
| } |
| } // end of parallel |
| return pos; |
| } |
|  |
| int |
| main() { |
| const int Ncpu = 2; |
|  |
| const std::size\_t count = 1000000; |
| std::vector<int> array(count); |
|  |
| for(std::size\_t i = 0; i < array.size(); i++) { |
| array[i] = i \* 2; |
| } |
|  |
| std::cout << "start ..." << std::endl; |
|  |
| boost::timer t; |
|  |
| int dummy = 0; |
| for(std::size\_t i = 0; i < array.size(); i++) { |
| dummy += nary\_search<int>(array, i, Ncpu); |
| } |
|  |
| std::cout << " -> " << t.elapsed() << std::endl; |
| } |