Communication Avoiding Conjugate Gradient Algorithms

Tyler Chen

One of the main drawbacks to Conjugate gradient in a high performance setting is .....

Communication bottlenecks in CG

Recall the standard Hestenes and Stifel CG implementation. In the below description, every block of code after a "**set**" must wait for the output from the previous block. Much of the algorithm is scalar and vector updates which are relatively cheap (in terms of floating point operations and communication). The most expensive computations each iteration are the matrix vector product, and the two inner products.

Algorithm. (Hestenes and Stiefel Conjugate Gradient)

procedure HSCG(A, b, x_0)
  set $r_0 = b - Ax_0, \nu_0 = \langle r_0, r_0 \rangle, p_0 = r_0, s_0 = Ar_0,$
    $a_0 = \nu_0 / \langle p_0, s_0 \rangle$
  for $k = 1, 2, ...$ **:**
    set $x_k = x_{k-1} + a_{k-1} p_{k-1}$
      $r_k = r_{k-1} - a_{k-1} s_{k-1}$
    set $\nu_k = \langle r_k, r_k \rangle$, **and** $b_k = \nu_k / \nu_{k-1}$
    set $p_k = r_k + b_k p_{k-1}$
    set $s_k = Ap_k$
    set $\mu_k = \langle p_k, s_k \rangle$, **and** $a_k = \nu_k / \mu_k$
  end for
end procedure

A matrix vector product requires $\mathcal{O}(\text{nnz})$ (number of nonzero) floating point operations, while an inner product requires $\mathcal{O}(n)$ operations. For many applications of CG, the number of nonzero entries is something like kn, where k relatively small. In these cases, the cost of floating point arithmetic for a matrix vector product and an inner product is roughly the same. On the other hand, the communication costs for the inner products are much lower.

matvec is usually sparse

inner products require "all reduce" i.e. collect information back from all the different processors/nodes would like to be able to overlap these as much as possible

Overlapping inner products

We would like to be able to *overlap* as many of the heavy computations as possible. However, in the current form, we need to wait for each of the previous computations before we are able to do a matrix vector product or an inner product.

Using our recurrences we can write,

$$s_k = Ap_k = A(r_k + b_k p_{k-1}) = Ar_k + b_k s_{k-1}$$

If we define the axillary vector $w_k = Ar_k$, in exact arithmetic using this formula for s_k will be equivalent to the original formula for s_k. However, we can now compute w_k as soon as we have r_k. Therefore, the computation of $\nu_k = \langle r_k, r_k \rangle$ can be overlapped with the computation of $w_k = Ar_k$.

These coefficient formulas seem to work better that CGCG..

Algorithm. (Chronopoulos and Gear Conjugate Gradient)

procedure CGCG(A, b, x_0)
  set $r_0 = b - Ax_0, \nu_0 = \langle r_0, r_0 \rangle, p_0 = r_0, s_0 = Ar_0,$
    $a_0 = \nu_0 / \langle p_0, s_0 \rangle$
  for $k = 1, 2, \ldots$ **:**
    set $x_k = x_{k-1} + a_{k-1} p_{k-1}$
      $r_k = r_{k-1} - a_{k-1} s_{k-1}$
    set $w_k = Ar_k$
      $\nu_k = \langle r_k, r_k \rangle,$ **and** $b_k = \nu_k / \nu_{k-1}$
    set $\eta_k = \langle r_k, w_k \rangle,$ **and** $a_k = \nu_k / (\eta_k - (b_k / a_{k-1}) \nu_k)$
      $p_k = r_k + b_k p_{k-1}$
      $s_k = w_k + b_k s_{k-1}$
  end for
end procedure

Algorithm. (Ghysels and Vanroose Conjugate Gradient)

procedure CGCG(A, b, x_0)
  set $r_0 = b - Ax_0, \nu_0 = \langle r_0, r_0 \rangle, p_0 = r_0, s_0 = Ar_0,$
    $w_0 = s_0, u_0 = Aw_0, a_0 = \nu_0 / \langle p_0, s_0 \rangle$
  for $k = 1, 2, ...$ **:**
    set $x_k = x_{k-1} + a_{k-1} p_{k-1}$
      $r_k = r_{k-1} - a_{k-1} s_{k-1}$
      $w_k = w_{k-1} - a_{k-1} u_{k-1}$
    set $\nu_k = \langle r_k, r_k \rangle,$ **and** $b_k = \nu_k / \nu_{k-1}$
      $\eta_k = \langle r_k, w_k \rangle,$ **and** $a_k = \nu_k / (\eta_k - (b_k / a_{k-1}) \nu_k)$
      $t_k = Aw_k$
    set $p_k = r_k + b_k p_{k-1}$
      $s_k = w_k + b_k s_{k-1}$
      $u_k = b_k u_{k-1}$
  end for
end procedure

fda

fdas