In the previous section, we discussed a particular technique of parallelization, that is, identifying when information regarding the inputs is known in advance, and inserting them into the circuit as soon as possible. In many cases, such as in factoring, the inputs (the control qubits $|p_j\rangle$ and the successive powers-of-two of the base a^{2^j} used for modular multiplication) are all known in advance, which allows us to create all $O(n)$ numbers in parallel at the beginning. However, this often comes at a cost in width, or number of qubits, which introduces another geometric constraint, when you consider a 2D circuit which you must eventually manufacture.

Instead of inserting inputs into a problem as you go along in a serial fashion, perhaps reusing qubits by uncomputing ancillae back to zero and making efficient use of space, now you have a state which is spread out across many qubits. This introduces a purely geometric communication problem of moving the information around this spread out state. As in the 2D factoring circuit in the previous example, the majority of the circuit size and circuit width ($O(n^6)$ over all the modular multipliers in use at any one time) is spent on moving the control qubits $|x_i \cdot y_j\rangle$ from their site of generation to the site of the z-numbers, which are spread out over a space $O(n^3)$ qubits long and requiring up to $O(n^4)$ as the maximum length of teleportation rail for any single qubit to be teleported. Since there are $O(n^2)$ control qubits, the total amount of operations is $O(n^6)$, even though the teleportations themselves happen concurrently in constant depth.

This often happens in establishing models, especially architectural models. Some constraint, such as nearest-neighbor operations occurring on a lattice, is established to approximate physical realism and more realistically model experimental settings. Most experimental implementations of quantum computing can only feasibly perform Hamiltonians which have locally-constrained interactions on a constant number of qubits. However, many experiments (cite the MUSIQC architecture here) no longer try to contain an entire circuit on a single physical substrate. Rather, many architectures believe that a future quantum computer will consist of many physical machines operating in parallel, with entanglement created by photon pairs exchanged by optical fiber and beamsplitters (citation needed here).

It is infeasible to imagine fabricating circuit board tracks on a silicon wafer for, for example, superconducting qubits, which is possibly many meters squared, and of which the majority consists of intermediate, reusable ancillae only useful for a few steps of constant-depth teleportation. It is much more likely that such a circuit would be split along boundaries into modules requiring the least amount of communication (similar to the classical problem of MAX-CUT, where the nodes represent qubits and the edges represent allowed interactions). These modules, which have minimum entanglement, can then be divided among communicating physical quantum computers.

Indeed, this is the amazing insight of recent work into constant-depth quantum circuits, so-called QNC^0, (cite BKP paper and Aram's distributed communication paper). Even though they are restricted to constant depth and polynomial width, they can still perform problems that are considered classically hard to do in constant-depth, such as factoring and even something as simple as parity and the majority function. Intuitively, if you have arbitrarily many small, fixed-size quantum computers that can communicate via nearest-neighbors (I actually don't know if this nearest-neighbor constraints on the larger module/QC level is valid, fact check this), you can compute the above functions in constant depth.

This represents the essence of the time-space tradeoff. In our previous architecture, we have sacrificed width, by allowing it to increase polynomially, in exchange for a lower depth (polylogarithmic). This would allow us to complete a circuit for factoring a reasonable-sized RSA key (4096 bits) in a matter of 8 days, versus the 427 years possibly needed by a 1D implementation, even one that used much fewer qubits. This would appear to be an advantage of our implementation, but qualified on the fact that it may take us many decades to scale up to widths of billions of qubits, or to sustain entanglement among millions of communication quantum computers long enough to run our 8 day computation. Indeed, given current rates of entangled photon pair generation (1 every few seconds, citation needed), such massive communication costs may be the major bottleneck.

Toward this even, we propose modifying the CCNTC to allow multiple communicating machines, and as another resource, we propose circuit modules, or the number of communicating sub-blocks needed. The circuit modules exist as nodes on a graph, again where allowed interactions are represented by edges and only nearest-neighbors can interact. Therefore, we can use this additional resource to model what is likely the most feasible model of scalable quantum computation.