Honeycomb-Structured Computational Interconnects and Their Scalable Extension to Spherical Domains

Joseph B. Cessna
Comp. Science, Math, & Engineering Program
University of California San Diego
La Jolla, CA, USA
jcessna@ucsd.edu

Thomas R. Bewley
Comp. Science, Math, & Engineering Program
University of California San Diego
La Jolla, CA, USA
bewley@ucsd.edu

ABSTRACT

The present paper is part of a larger effort to redesign, from the ground up, the best possible interconnect topologies for switchless multiprocessor computer systems. We focus here specifically on honeycomb graphs and their extension to problems on the sphere, as motivated by the design of special-purpose computational clusters for global weather forecasting. Eight families of efficient tiled layouts have been discovered which make such interconnects trivial to scale to large cluster sizes while incorporating no long wires. In the resulting switchless interconnect designs, the *physical proximity* of the cells created (in the PDE discretization of the physical domain) and the *logical proximity* of the nodes to which these cells are assigned (in the computational cluster) coincide perfectly, so all communication between physically adjacent cells during the PDE simulation require communication over just a single hop in the computational cluster.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*distributed networks, network topology*

General Terms

Design

1. INTRODUCTION

There are two paradigms for interconnecting processing elements in multiprocessor computer systems: switched and switchless.

Switched multiprocessor computer systems are the easiest to field and use in general-purpose applications, and are thus today the most popular. Fast cluster switching hardware has been developed by Infiniband, Myrinet, and Quadrics, and inexpensive ("commodity") switching hardware is available leveraging the standard gigabit ethernet protocol from Cisco. Unfortunately, in a switched computer system, the switch itself is a restrictive bottleneck in the system when attempting to scale to large cluster sizes, as messages between any two nodes must pass through the switch, and thus the throughput demands on the switch increase rapidly as the cluster size is increased. Nonuniform memory access (NUMA) architectures, first pioneered by Silicon Graphics, attempt to circumvent this quagmire by introducing a hierarchy of switches, thus allowing some of the "local" messages (that is, between two nodes on the same "branch" of a tree-like structure) to avoid passing through the full cascade of switches (that is, to avoid going all the way back to the "trunk"). This NUMA paradigm certainly helps, but does not eliminate the bottlenecks inherent to switch-based architectures.

Switchless multiprocessor computer systems, on the other hand, introduce a "graph" (typically, some sort of n-dimensional "grid") to interconnect the nodes of the system. In such a system, messages between any two nodes are relayed along an appropriate path in the graph, from the source node to the destination node. To accomplish such an interconnection in a beowulf cluster, relatively inexpensive PCI cards are available from Dolphin ICS [1]; however, the use of such hardware in today's high-performance clusters is fairly uncommon. The massively parallel high-performance Blue Gene design, by IBM, is a switchless three-dimensional torus network with dynamic virtual cut-through routing [2].

In the history of high-performance computing, switchless interconnect architectures have gone by a variety of descriptive names, including the 2D torus, the 3D torus, and the hypercube. Almost all such designs, including the IBM Blue Gene and the Dolphin ICS designs discussed above, imply an underlying Cartesian (that is, rectangular) grid topology in two, three, or $n > 3$ dimensions.

Quite recently, the startup SiCortex broke away from the dominant Cartesian interconnect paradigm, launching a novel family of switchless multiprocessor computer systems designed around the Kautz graph [9]. The Kautz graph is the optimal interconnect solution in terms of connecting the largest number of nodes of a given "degree" (that is, with a given number of incoming and outgoing wires at each node) for any prescribed maximum graph "diameter" (that is, the maximum number of hops between any two nodes in the graph). If one considers the wide range of possible graphs that may be used to interconnect a large number of computational nodes, the Cartesian graph may be identified as one extreme, with the simplest local structure possible but a poor graph diameter, whereas the Kautz graph may be considered the other extreme, with a complex logical structure that sacrifices local order but exhibits the optimal graph diameter.

In certain unstructured applications, the optimal graph diameter offered by the Kautz graph is attractive, though such systems become difficult to build as the cluster size is increased due to the intricate weave of long wires spanning the entire system.

Many problems of interest in high performance computing, however, have a regular structure associated with them. A prime example is the discretization of a partial differential equation (PDE). When distributing such a discretization on a switchless multipro-

SLIP'09, July 26–27, 2009, San Francisco, California, USA.

cessor computer systems for its parallel solution, one generally divides the domain of interest into a number of finite regions, or Voronoi cells, assigning one such cell to each computational node. An important observation is that such computations usually require *much* more communication between neighboring cells than they do between cells that are physically distant from one another. Thus, the practical effectiveness of proposed solutions to (i) the definition of the Voronoi cells, and (ii) the distribution of these cells over the nodes of the cluster (together referred to as the "load balancing problem") is closely related to both the physical proximity of the cells created in the PDE discretization and the logical proximity of the nodes to which these cells are assigned in the computational cluster. A graph with local structure, such as the Cartesian graph, can drastically reduce the average number of hops of the messages it must pass during the simulation of the PDE by laying out the problem in such a way that these two proximity conditions coincide; a graph without such local structure, such as the Kautz graph, does not admit an efficient layout which achieves this condition.

The present line of research thus considers alternative (noncartesian) graphs with local structure exploitable by PDE discretizations, while keeping to a minimum both (a) the number of wires per node [to minimize the complexity/expense of the cluster], and (b) the graph diameter [to minimize the cost of whatever multi-hop communication is required during the PDE simulation].

This particular paper is motivated by the needs presented by global weather forecasting problems defined over a sphere; note that some of the largest purpose-built computational clusters in the world are dedicated to this application. Loosely speaking, the present paper explores the best ways to put a fine honeycomb grid on a sphere, and then explores how to realize this discretization efficiently on an easily-scaled layout of computational hardware without using any long wires.

The work considered may be applied immediately at the system level. With the further development of appropriate hardware, it may also be applied at the board level or even the chip level. Other chip-level noncartesian interconnect strategies which have been investigated in the literature include the Y architecture and the X architecture. The Y architecture for on-chip interconnects is based on the use of three uniform wiring directions (0^o, 120^o, and 240^o) to exploit on-chip routing resources more efficiently than the traditional Cartesian (a.k.a. Manhattan) wiring architecture [4, 5]. The X architecture is an integrated-circuit wiring architecture based on the pervasive use of diagonal wires. Note that, compared with the traditional Cartesian architecture, the X architecture demonstrates a wire length reduction of more than 20% [11].

2. CARTESIAN INTERCONNECTS

Two criteria by which switchless interconnects are measured are cluster diameter and maximum wire length [6]. Loosely speaking, the former affects the speed at which information is passed throughout the graph, whereas the latter affects the cost of each wire used to construct the interconnect, as described further below.

Because each node can communicate directly only with its logical neighbors, we characterize information as moving in hops: it takes one hop for information to travel from a given node to its immediate neighbor, two hops for information to travel to a neighbor of a neighbor, etc. The diameter of a graph is the maximum number of hops between any two nodes in the graph. For example, Figure 1 illustrates a 1D Cartesian graph with a periodic connection. Each node can send information to its neighbor to the right, and receive information from its neighbor to the left. This type of connection is called a unidirectional link, because information can only flow in one direction. Many switchless clusters use bidirectional links, through which a node can both send and receive data.

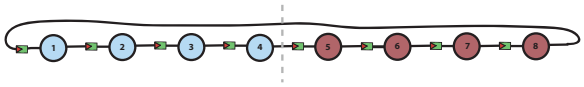

Figure 1: A simple 1D periodic Cartesian interconnect with unidirectional links. The diameter of this graph is seven hops. Note that a long wire is needed to make the periodic connection; the length of this wire increases as the cluster size increases.

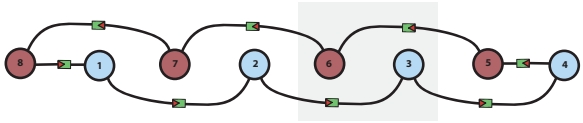

Figure 2: By folding the simple 1D interconnect of Figure 2 in half while keeping the same logical connection, the effect of the periodic connection may be localized. In this case, the longest wires only span the distance between two nodes in the folded structure, regardless of the number of nodes in the graph.

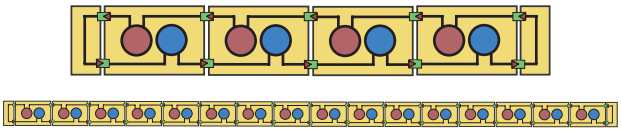

Figure 3: By identifying the local structure of the folded 1D interconnect of Figure 2, a tile may be designed that contains two nodes and four wires. This tile, together with simple end caps, may be extended to larger interconnects with the same topology. Here, we extend from 8 nodes (top) to 32 nodes (bottom).

The expense of the hardware required to complete a hop often increases quickly with the physical length of the wire between the nodes. To reduce this cost, it is thus desirable to minimize the maximum wire length. In Figure 1, the link connecting nodes 1 and N traverses N nodes, which makes scaling this layout to large N costly. The problem of long wires can be circumvented by folding the graph. By keeping the same logical connection, but folding the graph onto itself along its axis of symmetry, one can produce the graph shown in Figure 2. Here, the interconnect is identical to that of Figure 1 (and, thus, so is the graph diameter), but now the longest wire only spans the distance between two nodes, independent of N, thus facilitating scaling of the cluster to large N.

Noting the repetitive pattern in Figure 2, we identify a self-similar tile that can be used to build the interconnect. This tile is composed of two nodes and four wires (two sending and two receiving). Figure 3a illustrates how four of these tiles, along with simple end caps, can be combined to produce the original interconnect of Figure 1. An important feature of the tiled configuration is its scalability; note how it can be extended to much larger interconnects with the same topology, such as the 32 node graph in Figure 3b.

We now consider a four-connected periodic 2D Cartesian graph, known as a torus, in which each node has four unidirectional links, two for sending and two for receiving, as illustrated in Figure 4a. Similar to the 1D graph of Figure 1, the diameter of this 2D graph is six hops, but now interconnects 16 nodes instead of 8. The periodic connections of this 2D graph create many long wires that span the entire width of the interconnect. These long wires can be eliminated by folding the graph onto itself along both axes of symmetry.

From the folded 2D graph, we again identify local structure that facilitates tiling. The tiles for the 2D Cartesian torus contain four nodes and the associated communication links. Figure 5 shows how these tiles, together with simple end caps, may be assembled

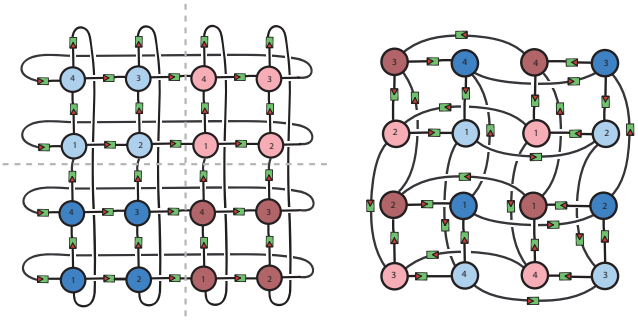

Figure 4: The idea of folding the interconnect to minimize the maximum wire length extends directly to higher dimensions. Here, a 2D periodic Cartesian graph (left) is folded onto itself in both directions of symmetry. Again, the longest wires only span the distance between two nodes in the folded structure.

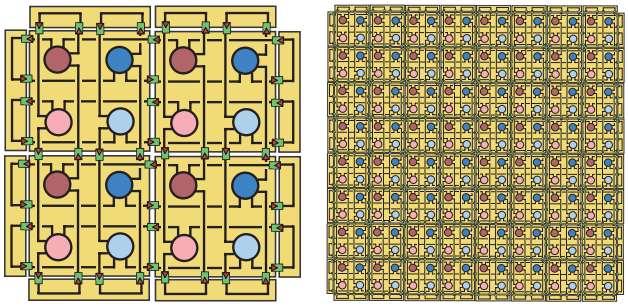

Figure 5: A four-node tile can be extracted from the interconnect of Figure 4b. This tile, combined with simple end caps, can be extended to larger interconnects with the same topology. Here, we extend from 16 nodes (left) to 256 nodes (right).

to produce the original 2D periodic Cartesian graph, and scaled to larger graphs of the same topology.

The three key steps illustrated by example in this section are:

(i) folding a graph to minimize the maximum wire length,
(ii) identifying repetitive local structure in the folded graph, and
(iii) defining a self-similar tiling to facilitate scaling.

The remainder of this paper extends these three steps to honeycomb graphs with a variety of useful periodic closures. The three-connected graphs so generated, in which each node is connected to an odd number of nearest neighbors, lack the symmetry required to configure an effective interconnect using unidirectional links. As a result, each link in the remainder of this discussion is intended to represent either a bidirectional link or a pair of unidirectional links (one in each direction). In §3.1, we consider the interconnects that arise from periodically connecting the honeycombl graph in the directions of the Cartesian unit vectors, much like the graphs of Figure 4, to produce a toroidal class of interconnects. In §3.2, we then examine the tilings that arise by periodically closing a honeycomb graph in three directions instead of two. Finally, §3.3 examines a variety of methods for wrapping a sphere with a (mostly) honeycomb graph.

3. HONEYCOMB INTERCONNECTS

Honeycomb (three-connected) graphs may be used in lieu of Cartesian (four-connected) graphs to create 2D interconnects with a significantly reduced number of wires (and, thus, a significantly reduced cost). We now show that such graphs can be developed with

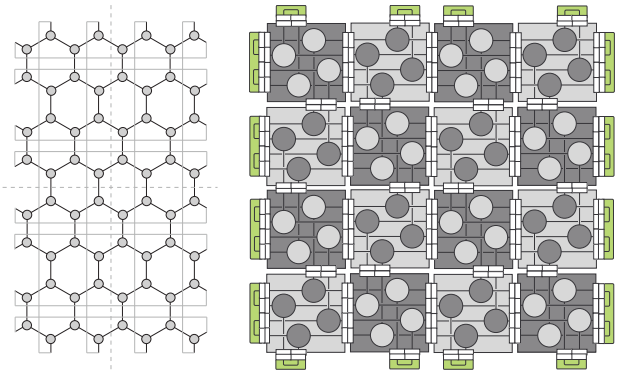

Figure 6: A 2D honeycomb grid is anisotropic about its Cartesian axes of symmetry. Nonetheless, periodic connections can be made in the directions of the Cartesian unit vectors (left). This closure produces a topology called a toroidal nanotube. By folding about the Cartesian axes, two distinct tiles can be identified to construct the interconnect (right). This interconnect has the same diameter the corresponding Cartesian torus with the same number of nodes, but uses 25% fewer wires.

essentially no increase in the overall complexity of the layout, and can easily be extended from toroidal closures, as discussed above, to triply-periodic closures, and then to spherical closures.

3.1 Toroidal closure

The simplest method for laying out a 2D periodic honeycomb graph is shown in Figure 6a. By making a periodic connection in the direction of one of the Cartesian unit vectors, we make a tubular topology similar to that of a carbon nanotube. This nanotube-like structure is then closed upon itself about its other axis of symmetry, forming a torus. Like the Cartesian torus, one can modify this closure by applying varying amounts of twist in one or both periodic directions before closing the graph. Such twists might prove useful in future applications, but for brevity are not considered further here.

Tiling the toroidal closure. As with the Cartesian torus, the periodic connections in the toroidal nanotube illustrated in Figure 6a create long wires that span the width of the entire graph, thus hindering scalability. To eliminate these long wires, a folding strategy is again used to reveal local structure and identify a tiling, as illustrated in Figure 6b. Unlike the Cartesian case (due primarily to the anisotropy of the topology with respect to the closure applied), we now define two distinct tiles, each with four nodes. However, the overall complexity of the tiling is on par with that of the Cartesian interconnect discussed previously. It is observed that the honeycomb tilings of the family illustrated in Figure 6 (with bidirectional links) have essentially the same diameter as the corresponding Cartesian tilings with the same number of nodes. Hence, by moving to a honeycomb topology, we develop a graph with the same diameter but only 3/4 of the wiring cost/complexity.

3.2 Triply-periodic closure

Although we can improve upon Cartesian interconnects with honeycomb graphs while still using a Cartesian closure strategy, as discussed above, it is more natural to select a closure for the honeycomb graph that better reflects its inherent symmetries. Towards this end, we now examine the triply-periodic closure of the plane honeycomb graph. This study forms the foundation upon which our study of spherical closures (§3.3) is based. Solutions to this

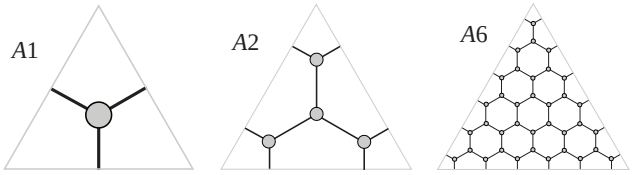

Figure 7: Three Class *A* structures (that is, honeycomb graphs on the equilateral triangle) with degree = 1, 2, and 6.

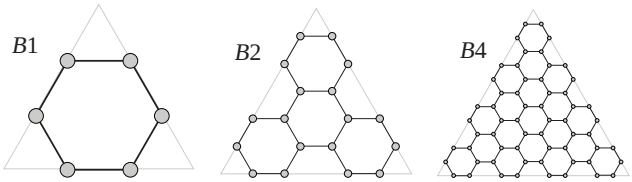

Figure 8: Three Class *B* structures with degree = 1, 2, and 4.

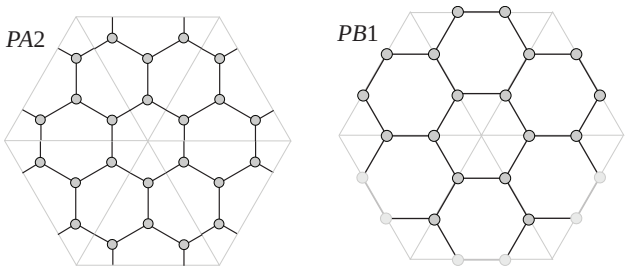

Figure 9: The two families of triply-periodic honeycomb interconnects. The *PA*∗ family is built from six Class *A* structures and has connected edge links whereas the *PB*∗ family is built from six Class *B* structures and has coincident edge nodes.

problem build from two distinct classes of honeycomb graphs on the equilateral triangle, denoted in this work as Class *A* and Class *B* structures:

A) The Class *A* structures place the midpoints of the links on the edges of the equilateral triangle, as illustrated in Figure 7. The degree of this structure is defined as the number of midpoints that lie on each edge of the triangle.

B) The Class *B* structures place the edges of the hexagons on the edges of the equilateral triangle, as illustrated in Figure 8. The degree of this structure is defined as the number of hexagons touching each edge of the triangle.

It is straightforward to join six Class *A* or Class *B* structures, as illustrated in Figures 7 and 8, to form a hexagon, as illustrated in Figure 9. The periodic connections on this hexagon are easily applied: in the case of Class *A*, the wires on opposite sides of the hexagon are connected; in the case of Class *B*, the nodes on opposite sides of the hexagon are taken to be identical (in both cases, moving orthogonal to each side of the hexagon, not diagonally through the center point). The resulting graphs are denoted *PA*∗ and *PB*∗, where the ∗ denotes the degree of the six Class *A* or Class *B* structures from which the triply-periodic honeycomb graph is built.

Tiling the triply-periodic closure. As in the previous examples, the triply-periodic graphs of Figure 9 can be tiled via folding about the three axes of symmetry and identifying the repetitive local structure of the folded graph. This process eliminates all long wires which grow as the graph size is increased. For Class *A*, the new tile so constructed, denoted Tile *E* in Figure 10, contains six nodes instead of four, leading to the tiled *PA*∗ family illustrated in

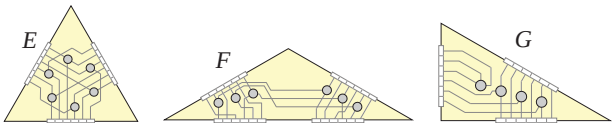

Figure 10: The three fundamental tiles, denoted *E*, *F*, and *G*, upon which the tilings of the triply-periodic (§3.2) and spherical (§3.3) closures of the honeycomb interconnect are based.

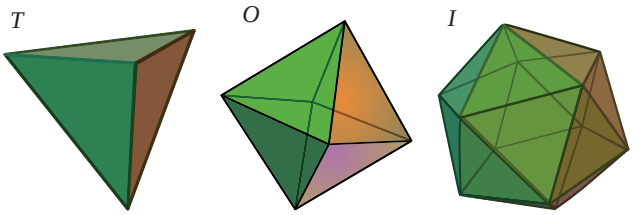

Figure 11: The three Platonic solids with triangular faces. From left to right: Tetrahedron (4 faces), Octahedron (8 faces), and Icosahedron (20 faces). The faces of these polyhedra can be gridded with either Class *A* or Class *B* triangular graphs (see Figures 7 and 8) to build tiled spherical interconnects based on the fundamental tiles introduced in Figure 10.

the first four subfigures of Figure 18. For Class *B*, the new tile so constructed contains 18 nodes; as illustrated in the last four subfigures of Figure 18, and for later convenience, we may immediately split this 18-node tile into three identical smaller tiles, denoted Tile *F* in Figure 10. The tilings are completed with simple end caps.

3.3 Spherical closure

We now discuss the most uniform techniques available to cover a sphere with a (mostly) honeycomb grid.

Note first that each *An* structure of Figure 7 has $V = n^2$ vertices (that is, nodes) and $E = 1.5n^2$ edges (that is, wires between nodes), whereas each *Bn* structure of Figure 8 has $V = 3n^2$ vertices and $E = 4.5n^2$ edges. If each corner of an *An* structure is joined with five other identical *An* structures, then each *An* structure contributes effectively $F = 0.5n^2$ faces (that is, hexagons) to the overall graph, whereas if each corner of a *Bn* structure is joined with five other *Bn* structures, then each *Bn* structure contributes $F = 1.5n^2$ faces to the overall graph. In both cases, we have $V - E + F = 0$, which is characteristic of a planar graph.

Euler's formula $V - E + F = 2$ relates the numbers of vertices, edges, and faces of any convex polyhedron. The upshot of Euler's formula in the present problem is that it is impossible to cover a sphere perfectly with a honeycomb grid. By this formula, any attempt to map a honeycomb grid onto the sphere will lead to a predictable number of "defects" (that is, faces which are not hexagons).

The three most uniform constructions available for generating a mostly honeycomb graph on a sphere are given by pasting a set of identical *An* structures of Figure 7, or *Bn* structures of Figure 8, onto the triangular faces of a Tetrahedron, Octahedron, or Icosahedron (see Figure 11); these three constructions form the basis for the remainder of this study. Note that this approach joins each corner of the structure selected with two, three, or four other identical structures, and leads to 4 triangular faces, 6 square faces, or 12 pentagonal faces embedded within an otherwise honeycomb graph. The resulting graph is easily projected onto the sphere. All graphs so constructed, of course, satisfy Euler's formula. Some of the graphs so constructed occur in nature as nearly spherical carbon molecules known as Buckyballs.

Degree	Symmetry and Class							
	Periodic		*Tetrahedral*		*Octahedral*		*Icosahedral*	
	A	*B*	*A*	*B*	*A*	*B*	*A*	*B*
1	6	18	4	12	8	24	20	60
2	24	72	16	48	32	96	80	240
3	54	162	36	108	72	216	180	540
4	96	288	64	192	128	384	320	960
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
k	$6k^2$	$18k^2$	$4k^2$	$12k^2$	$8k^2$	$24k^2$	$20k^2$	$60k^2$

Table 1: Number of nodes in a honeycomb interconnect. For the same symmetry and degree, the Class *B* topologies have three times as many nodes as Class *A*. Note also that *OA*1 is a cube, *IA*1 is a dodecahedron, and *IB*1 is a buckminsterfullerene.

Table 1 shows the number of nodes in these graphs as a function of their symmetry (*P*, *T*, *O*, or *I*), class (*A* or *B*), and degree (k). For each case, the number of nodes increases with k^2. For a given symmetry and degree, the Class *B* graph contains three times as many nodes as the Class *A* graph.

For general-purpose computing on switchless multiprocessor computer systems, spherical interconnects (§3.3) are not quite as efficient as planar interconnects (§3.2) with either toroidal closure or triply-periodic closure, as spherical interconnects have slightly higher diameters for a given number of nodes. However, as mentioned previously, spherical interconnects are poised to make a major impact for the specific problem of solving PDEs on the sphere, which is an important class of computational grand challenge problems central to a host of important questions related to global weather forecasting, climate prediction, and solar physics. In fact, some of the largest purpose-built supercomputers in the world are dedicated to such applications, and thus it is logical to design some computational clusters with switchless interconnects which are naturally suited to this particular class of applications.

Some of the graphs developed here lend themselves particularly well to efficient numerical solution of elliptical PDEs via the iterative geometric multigrid method; Figure 12 illustrates six graphs that may be used for such a purpose ($OA2^p$ for $p = \{0, 1, \cdots, 5\}$). Note that the Octahedral symmetry of these grids introduces a key property missing from the Tetrahedral and Icosahedral families of spherical interconnects: due to the square defect regions of the Octahedral graphs, a red/black ordering of points is maintained over the entire graph. That is, half of the points may be labeled as red and the other half labeled as black, with red points having only black neighbors and black points having only red neighbors. This ordering allows an iterative smoothing algorithm known as Red/Black Gauss Seidel to be applied at each sub step of the multigrid algorithm and facilitates remarkable multigrid convergence rates and efficient scaling on multiprocessor machines to very large grids.

Although the idea of building the interconnect of a multiprocessor computer system in the form of a honeycomb spherical graph is still in its infancy, people have known about and used such graphs to discretize the sphere in the numerical weather prediction community for many years; see, e.g., [3], [8], and [10]. In particular, the model developed by the Deutscher Wetterdienst (DWD) is an example of an operational meteorological code that implements the dual of a honeycomb graph with icosahedral symmetry (see [7]). These previous investigations have almost exclusively used graphs with Icosahedral symmetry, as such graphs undergo the least deformation when being projected from the polyhedron (on which they

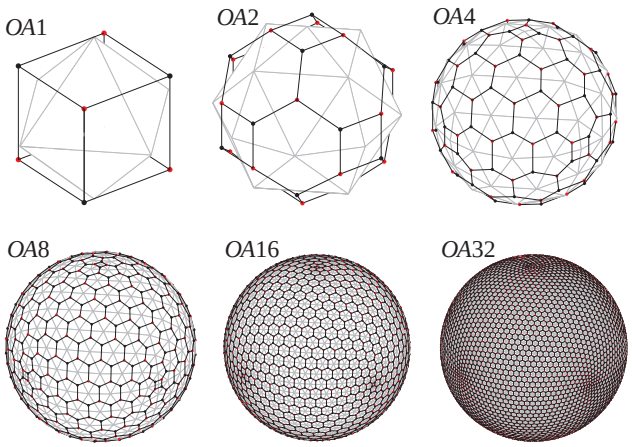

Figure 12: A representative family of spherical interconnects. All six graphs are members of the $OA2^p$ family (three of the square "defect" regions are visible in each graph). From top-left to bottom-right the degree is $2^0, 2^1, 2^2, 2^3, 2^4$, and 2^5. The thick black lines show the logical interconnect. The light grey lines illustrate the respective Voronoi cell for each node, indicating the region of the physical domain that each node is responsible for in a PDE simulation on a sphere.

are built) onto the sphere (where they are applied). Minimizing such graph deformation is beneficial for improving accuracy in numerical simulations. However, as mentioned above, our research indicates that Octahedral graphs, which admit a valuable red/black ordering not available in Icosahedral graphs, might, on balance, prove to be significantly better suited in many applications.

Tiling the spherical closure. As illustrated in Figure 12, spherical graphs are quite complicated; their practical deployment for large N would be quite difficult without a scalable tiling strategy. Thus, in a manner analogous to that used in §3.1 and §3.2, we now transform these graphs to discover local patterns and identify self-similar tilings that make such constructions tractable and scalable.

To illustrate the process, consider first the *TB*2 interconnect as a representative example, as depicted in Figure 13. Note first that we replace the "folding" idea used in §2, §3.1, and §3.2 with a "stretching" and "grouping" strategy. The stretching step may be visualized by imagining all links in the spherical graph under consideration as elastic, then imagining the transformation that would result from grabbing all of the nodes near one of the vertices of the polyhedron, or near the center of one of the faces of the polyhedron, and pulling them out to the side (while maintaining the connectivity) until the resulting stretched graph may be laid flat. The resulting spider web-like structure is known as a Schlegel diagram.

For the *TB*2 Schlegel diagram in the top-right of Figure 13, the outer edge of the diagram is a triangle that corresponds to the nodes near one of the vertices of the tetrahedron of the unstretched graph, and near the center of the diagram is a hexagon that corresponds to nodes near the middle of the opposite face of the tetrahedron.

The next step in tiling a spherical graph is to identify repetitive structure. In the Schlegel diagram of Figure 13, after considerable deliberation, we have grouped the nodes in sets of six. This grouping is not unique, but it is with the choice shown that a simple tiled structure reveals itself. In general, nodes that are approximately the same distance from the center of the Schlegel diagram tend to be lumped together. From the selected grouping, we can create the flowchart shown on the lower-left of Figure 13. In this flowchart, all obvious honeycomb structure is removed, but now the groups of

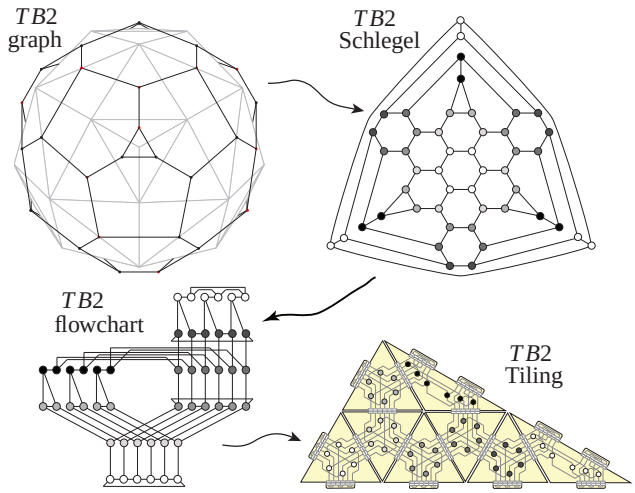

Figure 13: In order to determine a tiling for a given spherical interconnect graph (top-left), a Schlegel diagram (top-right) is first constructed by stretching and flattening the graph onto the plane while maintaining the same logical connections. From this, locally similar groups of nodes can be grouped into a revealing "flowchart" (bottom-left) that ultimately makes the tiling evident (bottom-right). In the case of $TB2$ illustrated here, with the exception of the top of each column of the flowchart, the nodes are connected identically to the $PA*$ case, leading to a tiling with the E tiles of Figure 10. This tiling is then completed with F tiles from Figure 10 along one side, together with the appropriate end caps.

paired nodes are brought together in what will eventually become their tiled form. Significantly, the logical structure of the flowchart (as well as the Schlegel diagram) is identical to the original spherical interconnect. Note that flow from bottom to top on the flowchart corresponds to in-to-out flow in the Schlegel diagram.

With the exception of the top set of nodes in each column of the flowchart, the connections between the other six sets of nodes match those of the $PA*$ case. That is, the first node in each group is connected to the first node in the neighboring groups, etc. As a result, we can use the same E tiles (see Figure 10) to construct this part of the graph as were used in the $PA*$ constructions. The remaining two sets of nodes can be incorporated using the F tile used in the $PB*$ constructions. After making simple end connections, the completed tiling is shown on the bottom-right of Figure 13. In this case, the overall tiling builds out to a right triangle.

For each interconnect family, we perform a similar procedure for the first few graph degrees until a pattern emerges, first stretching/flattening the graph to make a Schlegel diagram, then identifying repetitive local structure by grouping the nodes in an in-to-out flowchart of the Schlegel diagram, and finally defining a tiling to facilitate scaling. Though this was an involved process, by so doing, we have discovered the simple and easily scaled families of tilings depicted in Figures 19 - 21.

Interesting and surprising correlations exist between the various simple tilings which we have discovered. We first observe that each symmetry family shares a common overall shape: the $T**$ family builds out to right triangles, the $O**$ family builds out to parallelograms, and the $I**$ family builds out to "six-sided parallelograms".

By identifying an equivalence between a pair of E tiles and a pair of F tiles, as depicted in Figure 14, we are also able to identify interfamily relationships between the tilings as well. Applying this

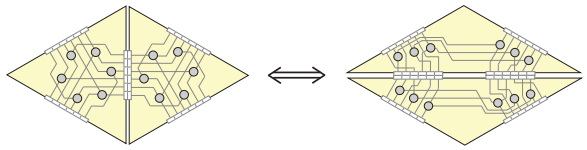

Figure 14: An equivalence between a pair of E tiles and a pair of F tiles. Typically, we use the E tiles whenever possible; however, the equivalence between these two pairings helps to unify the Class B tilings of the various different families.

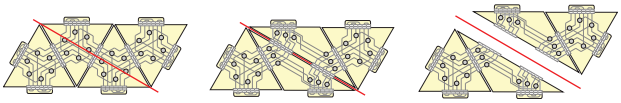

Figure 15: For any of the $OB*$ family of tilings (here, $OB1$ is shown), we can replace the E tiles along the shorter diagonal using the equivalence depicted in Figure 14. This allows us to identify the bifurcation axis that splits the tiling into two smaller tilings in the $TB*$ tilings of the appropriate degree.

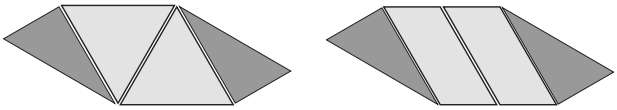

Figure 16: (left) The IBn graph may be subdivided into four smaller tilings [two TBn and two PBn] via a series of simple switches applied to the electrical connections between the tiles along the three diagonals shown; a similar subdivision to four smaller tilings [two TAn and two PAn] may also be accomplished with the IAn tiling after the transformation illustrated in Figure 14 is applied along the appropriate diagonals. (right) The IBn tiling, for n even, may also be subdivided a different way [to two TBn and two $OA(n/2)$]; a similar subdivision [to two TAn and two $OB(3n/2)$] may also be accomplished with the IAn tiling, for n even, after the transformation in Figure 14 is applied appropriately. Such subdivisions are useful for running four small jobs on the cluster simultaneously.

substitution along the shortest diagonal of the $OB*$ family, as illustrated in Figure 15, shows immediately how the $OB*$ tiling can be built up from two tilings of the $TB*$ family. Equivalently, the $OA*$ family can be built from two $TA*$ tilings of the appropriate degree.

Similarly, as illustrated in Figure 16, the $I**$ family of tilings can be built up from two $T**$ tilings and either two $P**$ tilings or two $O**$ tilings. Such subdivisions could be useful for running four smaller jobs on the cluster simultaneously without reconfiguration.

We have focused in §3.3 on applying the triangular Class A and Class B structures from Figures 7 and 8 to the triangular faces of three of the Platonic solids (Figure 11). However, other constructions are also possible. For instance, the truncated tetrahedron (Figure 17) has only hexagonal and triangular sides, and thus can be covered with triangular Class A or Class B structures on its triangular faces and six Class A or Class B structures (see Figure 9) on its hexagonal sides. The resulting Class A graphs have $28k^2$ nodes, and the resulting Class B graphs have $84k^2$ nodes (cf. Table 1). These graphs each have 12 pentagonal faces embedded within an otherwise honeycomb graph, just like the icosahedron; however, one projected out to the sphere, they are slightly less uniform and do not admit a red/black ordering of points. Another alternative construc-

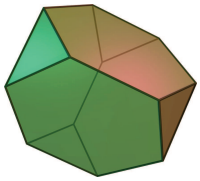

Figure 17: The truncated tetrahedron, the only Archimedean solid with only triangular and hexagonal sides.

tion places two $PA*$ of $PB*$ graphs on top of one another, connected along the edges, projecting the top onto the norther hemisphere and the bottom to the southern hemisphere. These graphs each have six square faces embedded within an otherwise honeycomb graph, just like the octahedron; however, they are significantly less uniform, though they do admit a red/black ordering of points. It is currently unclear whether or not such alternative constructions have any significant advantages.

4. SUMMARY

The present line of research considers the topology of the interconnection of processing elements in switchless multiprocessor computer systems. The design of such switchless interconnects is a problem that has been considered for decades; however, the question of scalability of such designs is of heightened importance today, as modern computer systems take parallelization to new levels. Note that the three fastest computer systems in the world today each has hundreds of thousands of nodes, whereas typical computer clusters in academic and industrial settings are today growing from thousands to tens of thousands of nodes. It is for this reason that, we believe, a revisiting of the topology used in switchless multiprocessor computer systems is in order, and special-purpose topologies for clusters built for special-purpose applications, such as global weather forecasting, are warranted.

The present paper focused on the optimal ways to cover a sphere with a fine honeycomb grid. Euler's formula leads to a predictable number of "defects" (non-hexagons) within the otherwise honeycomb grid; the three choices that may be developed with maximal uniformly incorporate 4 triangles (distributed at the corners of a Tetrahedron), 8 squares (distributed at the corners of an Octahedron), or 12 pentagons (distributed at the corners of an Icosahedron) into the otherwise honeycomb grid. The present work systematically examined all three of these cases (denoted $T**$, $O**$, and $I**$), and discovered two convenient families of tilings for each case (denoted $*A*$ and $*B*$). These tilings are repetitive structures, which greatly eases their scaling to large cluster sizes ($**n$ for $n \gg 1$), and contain no long wires, thereby facilitating fast link speeds and reduced cluster cost and complexity even as the cluster size scales to hundreds of thousands of nodes.

In the resulting switchless interconnect designs, the *physical proximity* of the cells created (in the PDE discretization on the sphere) and the *logical proximity* of the nodes to which these cells are assigned (in the computational cluster) coincide perfectly, so all communication between physically adjacent cells during the PDE simulation require communication over just a single hop in the computational cluster. Such an interconnect should provide near-perfect scaling with cluster size in operational problems; that is, refining the overall grid by a factor of n while also increasing the overall cluster size by a factor of n will result in an essentially unchanged execution speed. Such scaling is the holy grail of parallel computing, and is enabled in the present case by the careful design of a computational interconnect topology that is well matched to the physical problem to which the computational cluster is dedicated.

5. REFERENCES

[1] The Dolphin SCI interconnect. White paper, Dolphin Interconnect Solutions, 1996.

[2] N. R. Adiga, M. A. Blumrich, D. Chen, P. Coteus, A. Gara, M. E. Giampapa, P. Heidelberger, S. Singh, B. D. Steinmacher-Burow, T. Takken, M. Tsao, and P. Vranas. Blue Gene/L torus interconnection network. *IBM Journal of Research and Development*, 49(2/3):265, 2005.

[3] J. R. Baumgardner and P. O. Frederickson. Icosahedral discretization of the two-sphere. *SIAM J. Numer. Anal.*, 22(6):1107–1115, 1985.

[4] H. Chen, C.-K. Cheng, A. B. Kahng, I. I. Mandoiu, Q. Wang, and B. Yao. The Y architecture for on-chip interconnect: Analysis and methodology. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 24(4):588–599, 2005.

[5] H. Chen, B. Yao, F. Zhou, and C.-K. Cheng. The Y-architecture: yet another on-chip interconnect solution. In *Proceedings of the 2003 Conference on Asia South Pacific Design Automation*, pages 840–847, 2003.

[6] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Elsevier, Burlington, 2004.

[7] D. Majewski, D. Liermann, P. Prohl, B. Ritter, M. Buchold, T. Hanish, G. Paul, and W. Wergen. The operational global icosahedral-hexagonal gridpoint model GME: Description and high-resolution tests. *Monthly Weather Review*, 130:319–338, 2002.

[8] R. Sadourny, A. Arakawa, and Y. Mintz. Integration of the nondivergent barotropic vorticity equation with an icosahedral-hexagonal grid for the sphere. *Monthly Weather Review*, 96:351–356, 1968.

[9] L. C. Stewart and D. Gingold. A new generation of cluster interconnect. White paper, SiCortex, Inc., 2006.

[10] G. R. Stuhne and W. R. Peltier. New icosahedral grid-point discretizations of the shallow water equations on the sphere. *Journal of Computational Physics*, 148:23–58, 1999.

[11] S. L. Teig. The X architecture: Not your father's diagonal wiring. In *Proceedings of the 2002 International Workshop on System-Level Interconnect Prediction*, 2002.

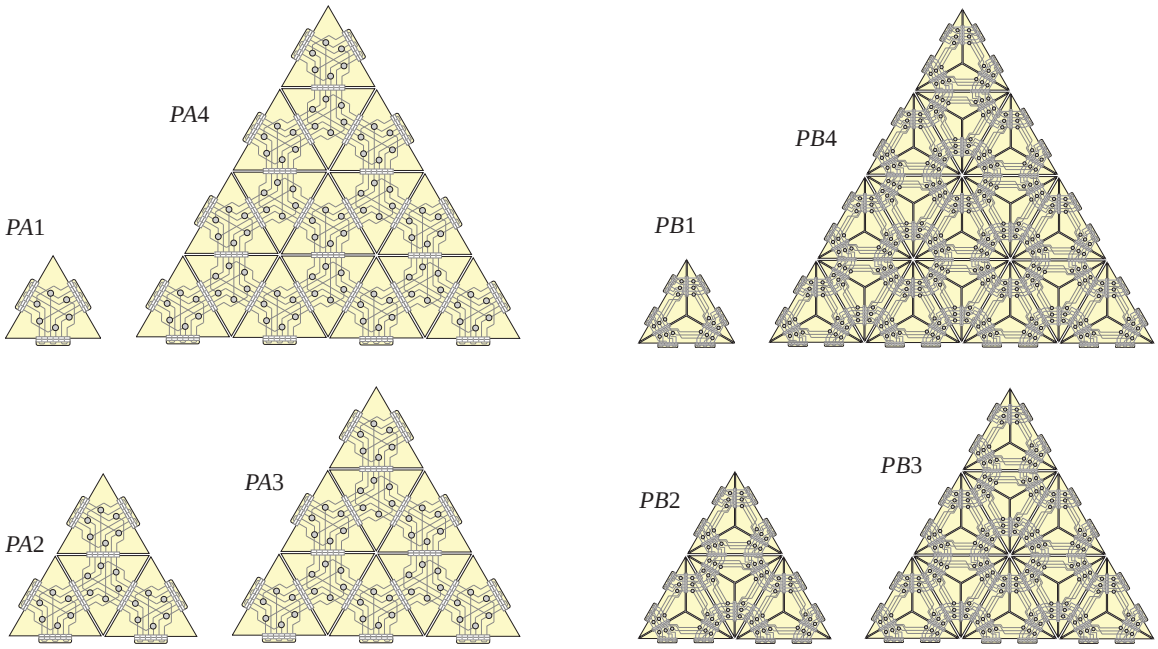

Figure 18: The first four members of the $PA*$ and $PB*$ families of interconnects; each builds out to an equilateral triangle. Note that the $PA*$ construction illustrated here is built using E tiles, whereas the $PB*$ construction is built using F tiles (see Figure 10).

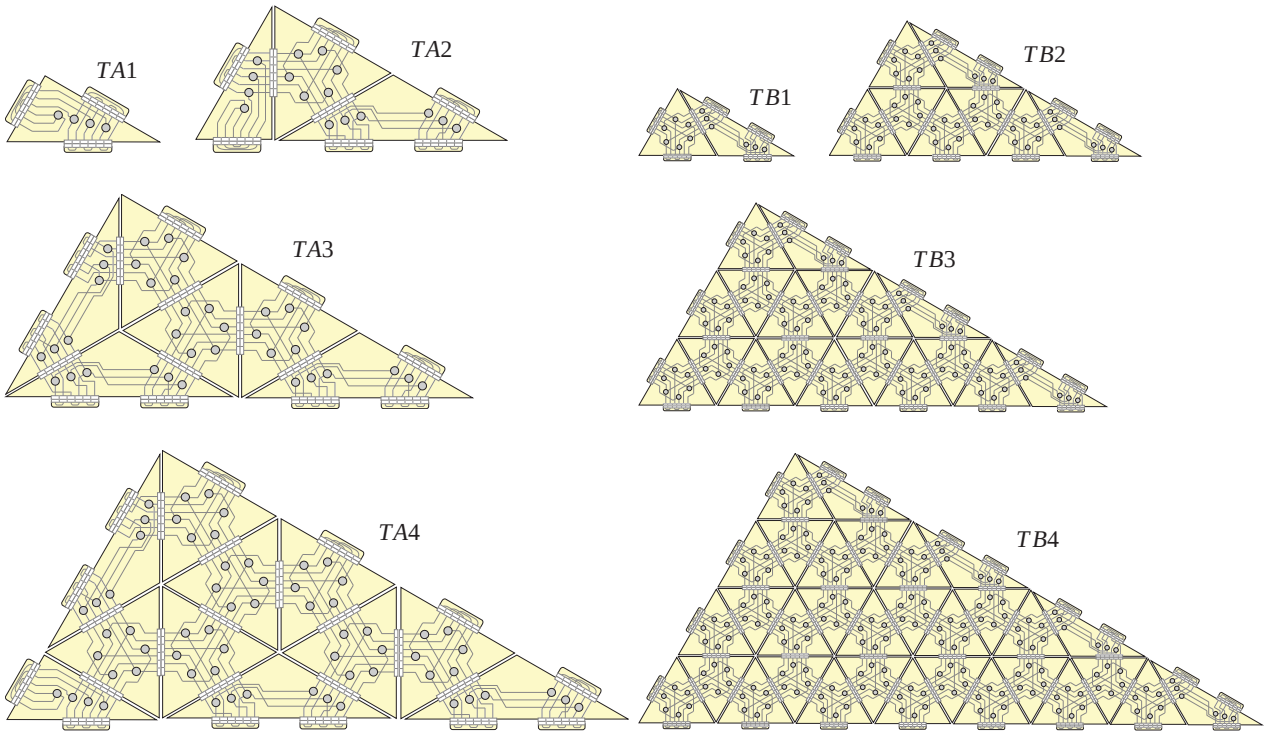

Figure 19: The first four members of the $TA*$ and $TB*$ families of interconnects; each builds out to a right triangle.

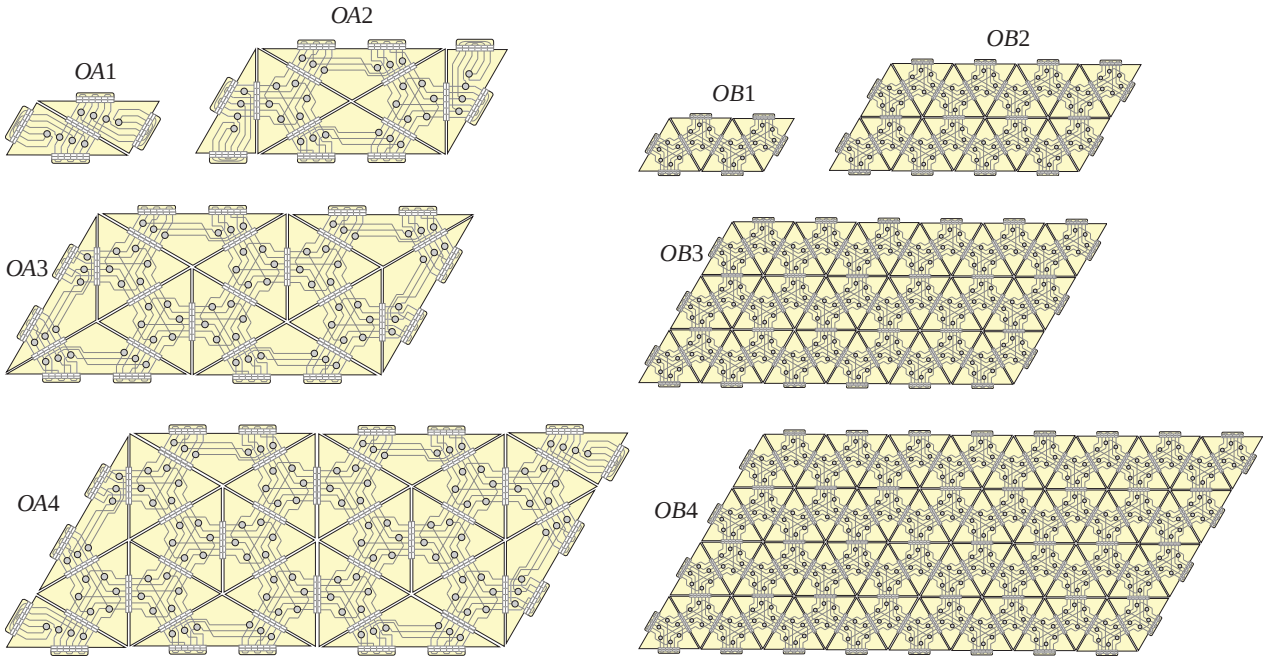

Figure 20: The first four members of the $OA*$ and $OB*$ families of interconnects; each builds out to a parallelogram. Both families may be constructed by connecting two $TA*$ or $TB*$ tilings along the longest leg, as described in Figure 15.

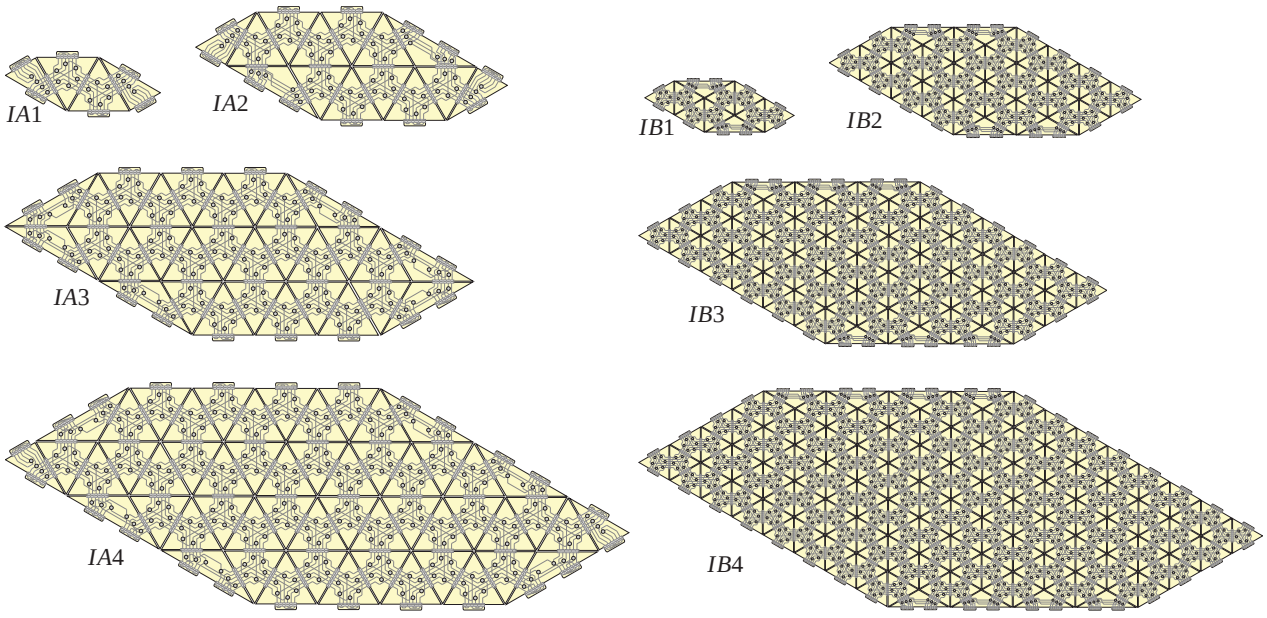

Figure 21: The first four members of the $IA*$ and $IB*$ families of interconnects; each builds out to a "six-sided parallelogram". Both families may be constructed by connecting two $T**$ and two $P**$ tilings, or two $T**$ and two $O**$ tilings, as described in Figure 16.