Approximate Graph Products

Wilfried Imrich

TBI Winterseminar
Bled, Slovenia

February 20, 2007

The main motivation is the potential role of approximate graph products for the definition and recognition of characters in evolutionary science and bioinformatics. The investigation of approximate graph products (AGP) is a new field, essentially initiated by Peter Stadler.

Problems are still very intuitively formulated. In this talk we consider the following version:

Given a graph G, find a product $G_1 \times G_2 \times \cdots \times G_k$ such that G can be obtained from $G_1 \times G_2 \times \cdots \times G_k$ by addition or deletion of just a few vertices and edges.

The problem is NP-complete in general if one asks for optimal solutions, but polynomial if the number of edges and vertices involved is bounded by an integer k.

There are similarities to canonical isometric embeddings of graphs into Cartesian products as introduced by Ron Graham in computer science and to H. J. Bandelt's application of median networks (retracts of hypercubes and Hamming graphs) in the phylogeny of mitochondrial DNA.

The original problem, as formulated by Peter Stadler, pertains to oriented graphs with respect to the direct product of graphs.

This is a very difficult problem. It is considerably easier for

the direct product of undirected graphs

the strong product of undirected graphs

and best understood for

the Cartesian product.

Since the graphs in question will be large, we need fast algorithms for the recognition of AGP, almost linear if possible.

Consider the recognition complexities for products. Given a connected graph on m edges and n vertices, the complexity of representing it as a product of indecomposable factors is

O(m) for the Cartesian product

and at least O(n^5) for the strong and the direct product. (For the direct product additional connectivity conditions are needed to ensure termination of the algorithm.)

For the direct product of oriented graphs large classes are known that can be treated by polynomial algorithms, but no complete answer is yet known.

It seems that the search for fast algorithms for the recognition of AGPs leads to new algorithms that improve the recognition complexities of the strong and the direct product and maybe to a new and simpler linear algorithm for the Cartesian product.

In the case of the strong and the direct product the resulting algorithm method is could be called a fast parametrized algorithm.

Now this talk becomes more concrete. It continues with the definition of the Cartesian product and two examples of approximate Cartesian products.

Then comes the strong product, an example of an approximate strong product, and an indication how one can recognize it.

An idea to recognize strong products as such and an estimate of the complexity.

This is followed by a treatment of the examples for approximate Cartesian products and an outlook on further investigations.

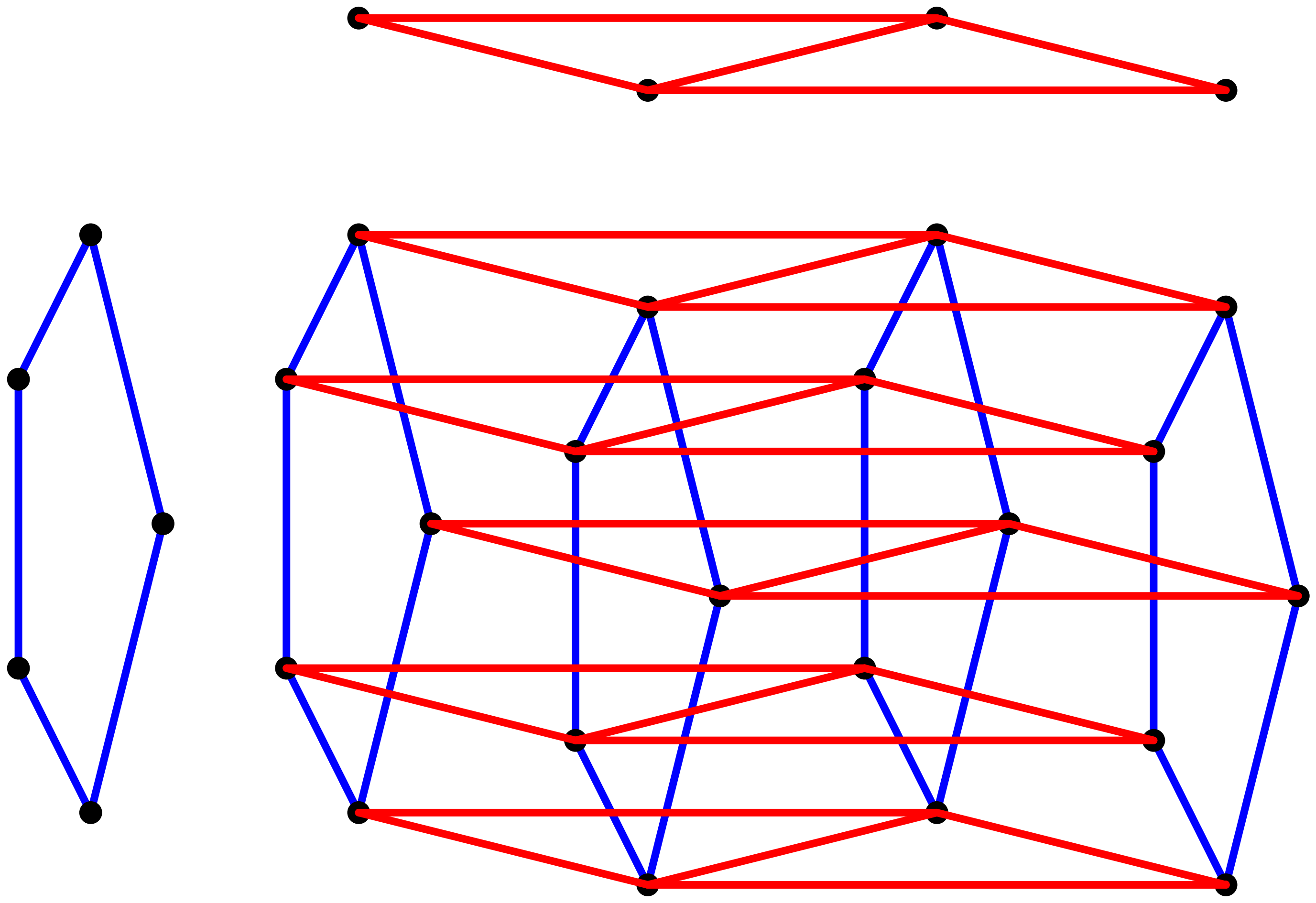
Definition of the Cartesian product $G \Box H$

$$V(G \Box H) = V(G) \times V(H),$$

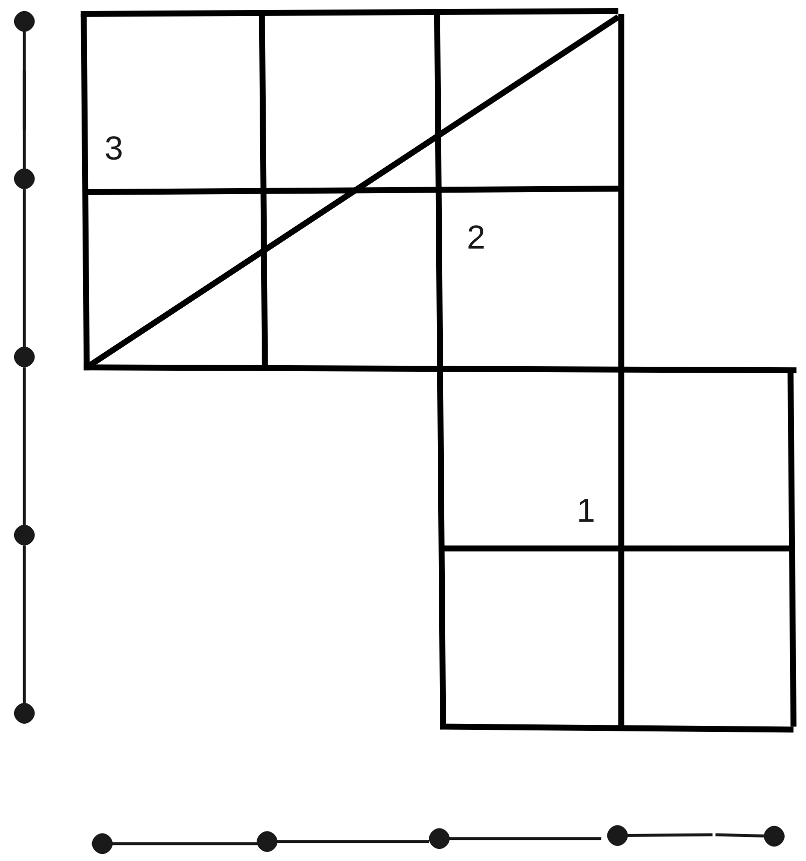
$E(G \Box H)$ that is the set of all pairs $[(u, v), (x, y)]$

where either $u = x$ and $[v, y] \in E(H)$ or $[u, x] \in E(G)$ and $v = y$

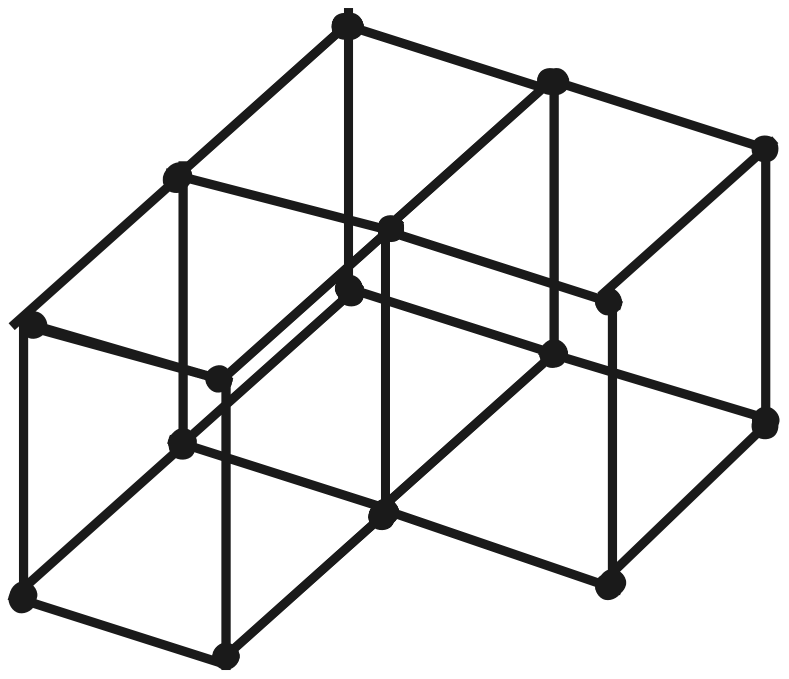
See the example on the next page:

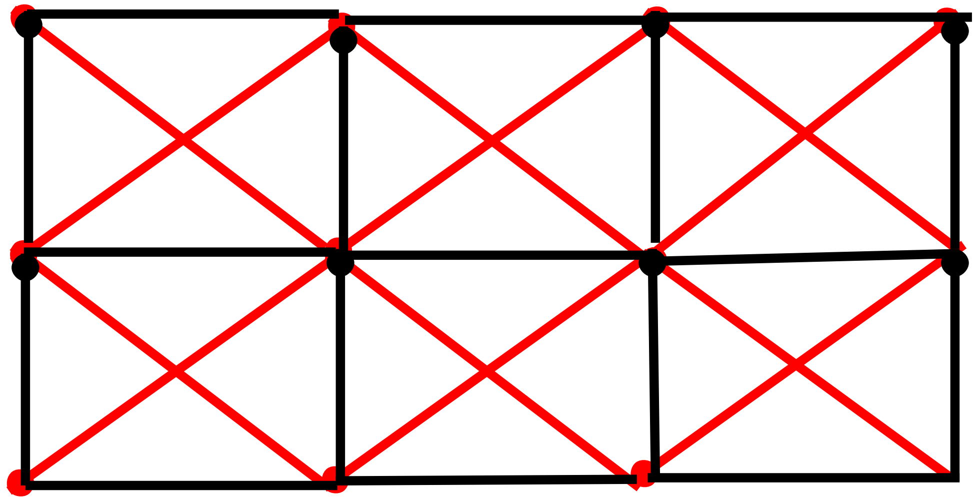


First example of an approximate Cartesian product

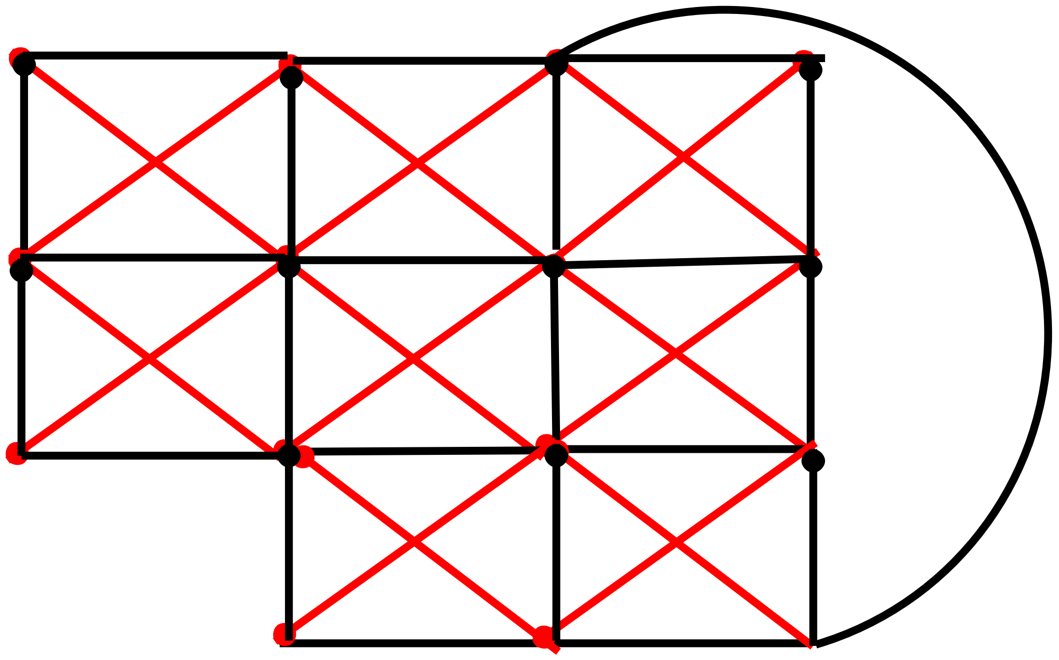


Second example of an approximate Cartesian product

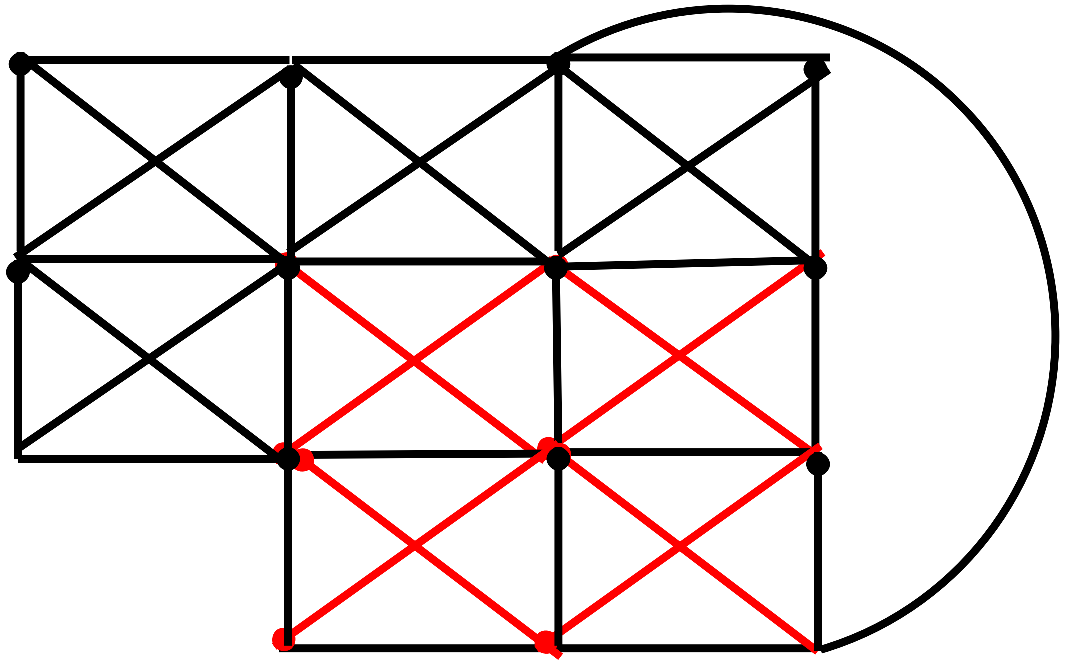




Example of an strong product



Example of an approximate strong product



Idea to recognize it as an approximate strong product

New algorithm for the recognition of strong products

1. Every neighborhood of a vertex is a neighborhood. Factor all of them.

2. Successively combine factorizations of neighborhoods with intersections that contain at least two vertices.

3. Check whether this yields a factorization of the given graph. Reduce the number of factors if necessary.

(For details see the talk by Werner Klöckl.)

Complexity of this approach if the degrees of G are bounded by d:

Every neighborhood can be factored in $\mathrm{O}(d^5)$ time. If G as n vertices Step 1 can be done in $\mathrm{O}(nd^5)$ time.

Steps 2 and 3 can be effected most likely in $\mathrm{O}(m) = \mathrm{O}(nd)$ time, thus $d^4\mathrm{O}(m)$ is the bound.

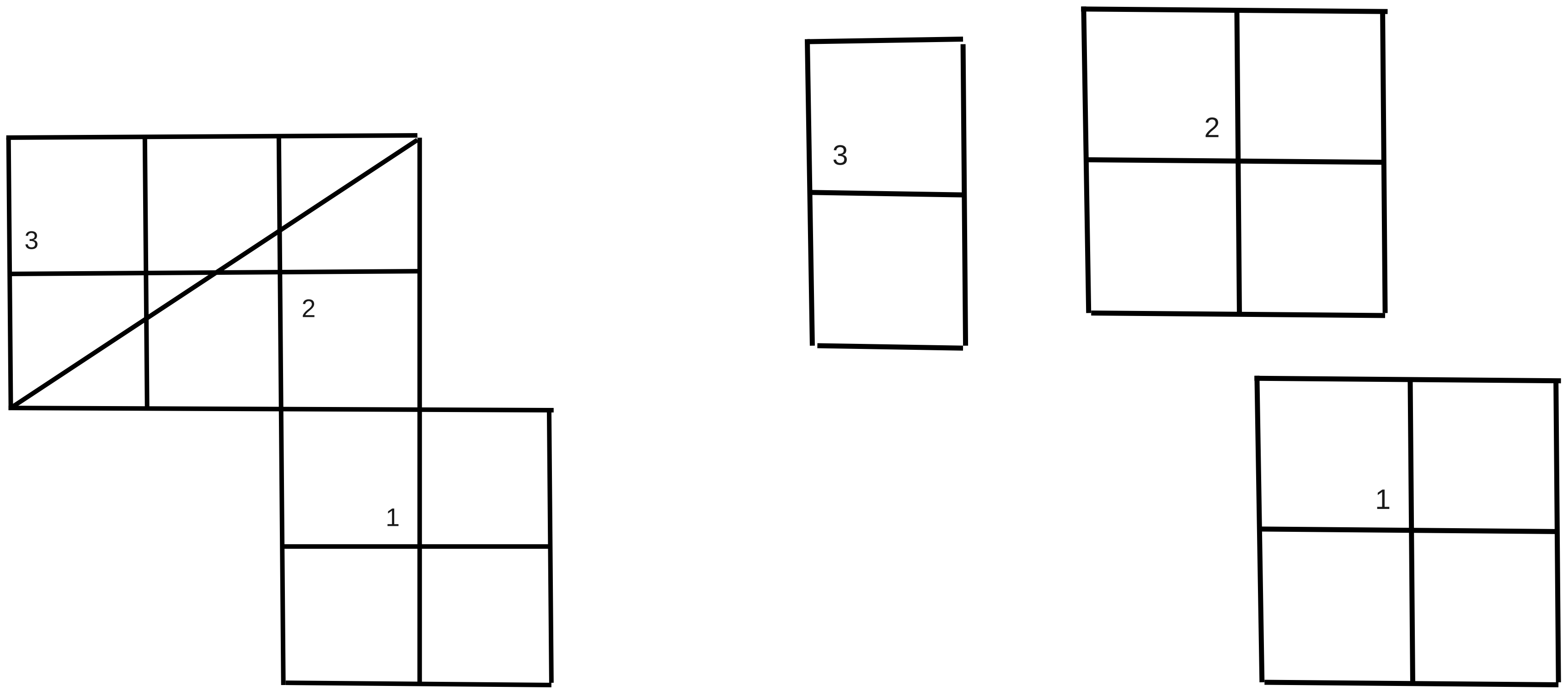
This compares to n^5 in general.

Essentially the same bound should apply for the recognition of approximate strong (and direct) products.

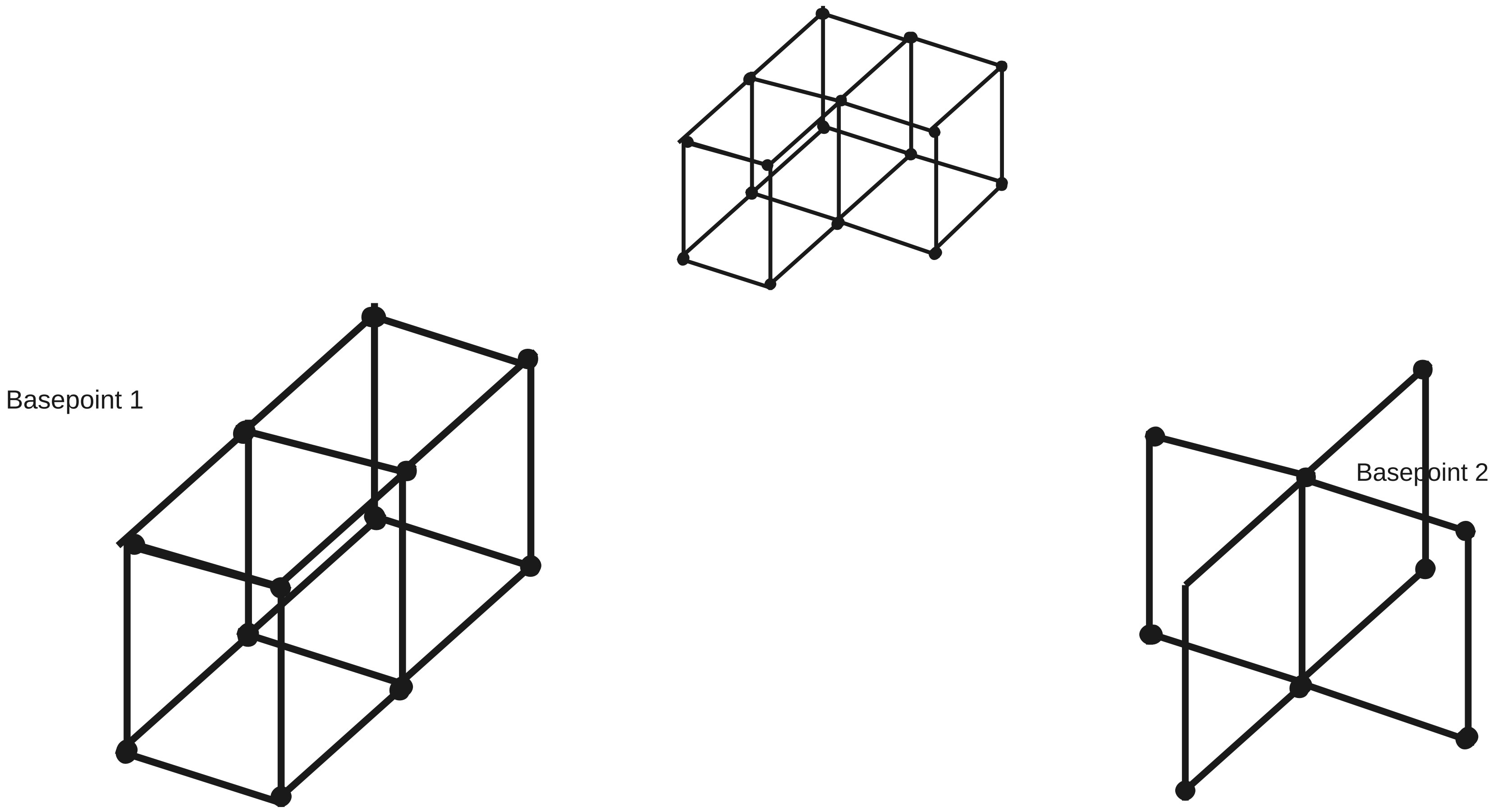
With the Cartesian product it is more difficult to define subproducts that can be used to cover the graph. One way is to use subproducts generated by the edges incident with every given vertex. They can be found in linear time each.

See the examples on the next slides.

Then the same procedure is followed as in the case of the strong product.



Partial products to the first example of an approximate Cartesian graph product.



Partial products to the second example of an approximate Cartesian graph product.

Conclusion

We expect to be able to find algorithms of low complexity for the recognition of approximate products for the Cartesian, the strong and the direct product of undirected graphs, and for a large class of oriented graphs in the case of the direct product.

The results will in general not be optimal, but good in a certain sense and normed (as the canonical isometric embedding of graphs into Cartesian products).

"We" refers to Peter Stadler in Leipzig, me in Leoben, and those members of our teams who work in this area. Currently these are Clemens Brand, Werner Klökl, Tomas Kupka, Wilfried Imrich in Leoben and Hellmuth Marc and Peter Stadler in Leipzig. We also appreciate the support by Christoph Flamm from Vienna.