Cube Connected Cycles Based Bluetooth Scatternet Formation

Marcin Bienkowski[1,⋆], André Brinkmann[2], Miroslaw Korzeniowski[1,⋆],
and Orhan Orhan[1]

[1] International Graduate School of Dynamic Intelligent Systems,
University of Paderborn, Germany
`{young, rudy}@upb.de`, `orhan@hni.upb.de`
[2] Heinz Nixdorf Institute, University of Paderborn, Germany
`brinkman@hni.upb.de`

Abstract. Bluetooth is a wireless communication standard developed for personal area networks (PAN) that gained popularity in the last years. It was designed to connect a few devices together, however nowadays there is a need to build larger networks. Construction and maintenance algorithms have great effect on performance of the network. We present an algorithm based on Cube Connected Cycles (CCC) topology and show how to maintain the network so that it is easily scalable. Our design guarantees good properties such as constant degree and logarithmic dilation. Besides, the construction costs are proven to be at most constant times larger than any other algorithm would need.

1 Introduction

In this paper we address the problem of network topology construction and maintenance for a wide variety of networks. We require any two nodes to be able to build a bidirectional communication link; for radio networks this can be achieved by placing all the nodes within the communication range. Our topology has a very low requirement for the maximum degree of a node. It is sufficient if the node is capable of communicating with 7 neighbors simultaneously.

The requirements above make the Bluetooth protocol [1] a perfect candidate for our network design. Bluetooth is one of the most recent wireless communication standards developed for Personal Area Networking. Its specification assigns roles of *masters* and *slaves* to nodes. The structure consisting of one master and up to 7 active slaves connected to it is called a *piconet*. Each piconet has a specific frequency-hopping channel which is controlled by its master. For efficiency reasons it is profitable to minimize the number of masters (and thus the number of piconets) and connect two masters not directly, but through a slave, to which we refer later as a *bridge*. Such connection of piconets by bridges can

---

[⋆] Partially supported by DFG-Sonderforschungsbereich 376 “Massive Parallelität: Algorithmen Entwurfsmethoden Anwendungen”.

P. Lorenz and P. Dini (Eds.): ICN 2005, LNCS 3420, pp. 413–420, 2005.

establish a large network structure called *scatternet*. Furthermore, the frequency hopping mechanism used by Bluetooth makes the situation, in which a bridge participates in more than two piconets, very undesirable, since the probability of collision between its masters grows very quickly.

An important property of a network is the possibility to maintain a simple routing scheme in it. Neither large routing tables nor long lasting path-finding routines should be used due to bounded network bandwidth and memory of the devices. Last but not least, dynamic scalability of the network should be taken into consideration. This means that nodes can join and leave the network at their convenience without loosing the mentioned characteristics.

In this paper we present a topology which has all the properties mentioned above. We start from the theoretical Cube-Connected-Cycles structure (CCC) [2] and we model it using Bluetooth devices. Each node in the theoretical structure is simulated by a Bluetooth master. Further, if we have a communication link between two nodes in the theoretical structure, we join the two corresponding masters by a bridge. Since in CCC each node has a degree of 3, each master will have 4 spare links which can be used for connecting additional slave nodes.

Among the networks with constant degree, our structure has asymptotically the best possible dilation of $\mathcal{O}(\log n)$; the constant hidden in the $\mathcal{O}$ notation is small. The scalability limits are set by the frequency hopping scheme used by Bluetooth protocol rather than by our topology. The maintenance cost is also optimal. We prove that for any sequence of nodes joining and leaving our network, the cost of our algorithm is at most 18 times larger than the cost of the optimal offline algorithm for the same sequence.

2 Scatternets: Related Work

The problem of scatternet formation for Bluetooth has been intensively studied in the last few years. The proposed algorithms can be categorized into two broad clases. The first group includes those that assume that all devices are in communication range of each other. The algorithms from the second group form a connected network also when this condition is not fulfilled. Due to space limitations we do not go into details for the second group. Check [3, 4, 5, 6] for more information.

Formations for Devices In Range. One of the earliest scatternet formation algorithm studied by Salonidis et al [7] is the Bluetooth Topology Construction Protocol (BTCP), which works only for at most 36 nodes. For a larger number of nodes it proposes a scheme that does not build a fully connected scatternet. Ramachandran et al [8] give two distributed algorithms (one randomized and one deterministic) which build optimal topologies consisting of stars. The issue of choosing bridges to connect the stars is left open. Baatz et al [9] propose a scheme based on composing the topology of k 1-factors (a 1-factor is a graph of maximum degree at most 1, i.e. consisting of independent edges). In each 1-factor one node of an edge is treated as a master and the other as a slave. This topology has an advantage of having multiple active piconets at the same time even if there is overlap between them. However, the roles of masters and slaves are distributed equally which is not desirable for scatternets. The tree scatternet formation (TSF) [10] is a self repairing structure, which organizes nodes into a tree. It allows nodes to arrive and leave arbitrarily. The tree structure guarantees that there are no loops in the network and thus that routing between any pair of nodes is unique. It succeeds in minimizing the number of piconets in the network but is not suitable for larger networks due to high delays in communication. Wang et al [11] define an algorithm called *Bluenet* which aims at constructing a random connected graph. The main disadvantage of this topology is that it lacks any structure which would enable simple routing. Lin et al introduce BlueRing[12] in which the scatternet is based on a ring structure. The architecture has a simple routing and is easy to maintain, however it is only scalable up to a medium sized networks (50-70 nodes). It is unusable for larger networks because of an average dilation and congestion being linear in the size of the network. One of the most advanced approaches in the design of scatternets is BlueCube[13] which proposes a d-dimensional hypercube as a theoretical basis of the network formation. It has a logarithmic dilation, but is only defined for a certain number of nodes. Since the degree of a Bluetooth node is limited to 7, this places also an upper bound on d, limiting the number of nodes in the network to approximately 2^7. The only truly scalable solution we are aware of is proposed in [14], where a network of constant degree and polylogarithmic diameter is constructed. The network is based on a backbone that enables routing based on virtual labeling of nodes without large routing tables or complicated path-discovery methods. The scheme is fully distributed and dynamic in the sense that nodes can join and leave the network at any time.

3 Building and Maintaining Large Scale Bluetooth Scatternets

Our approach is based on a network topology called Cube Connected Cycles. We consider this topology on the basis of the graph theory and adjust it to the Bluetooth specification. First we give a theoretical definition of the topology and show how it can be implemented using Bluetooth devices. Then we present a maintenance algorithm for Bluetooth scatternet based on CCC topology. The algorithm changes the structure instantly when nodes are joining or leaving the system and assures that the number of changes is constant in each step. In the full version of the paper we present another algorithm, which tries not to change the topology as long as possible; the resulting topology updates are large but happen very rarely. The amortized number of changes is even lower than in the case of the smooth maintenance scheme.

Cube Connected Cycles Topology.

Definition 1. *The d-dimensional Cube Connected Cycles network has $d \cdot 2^d$ nodes. The nodes are represented by two indices (i, j), where $0 \leq i < d$ and $0 \leq j < 2^d$. The connectivity is:*

$$(i,j) \rightarrow \begin{cases} (i, j \oplus 2^i) & 0 \leq i < d, 0 \leq j < 2^d \\ ((i \pm 1) \mod d, j) & 0 \leq i < d, 0 \leq j < 2^d \end{cases}$$

where $\oplus$ represents the bitwise xor operation. The first set of edges are the cube edges; the second set of edges are the cycle edges.

Observation 1. *The d-dimensional Cube Connected Cycles network has the following properties:*

1. *The number of nodes is $n = d \cdot 2^d$.*
2. *The degree of each node is 3 (or smaller for $d \leq 2$).*
3. *The number of edges is $m = \frac{3}{2} \cdot n = 3 \cdot d \cdot 2^{d-1}$ (or smaller for $d \leq 2$).*
4. *For any two nodes a and b we can compute a path from a to b of length at most $3 \cdot d$.*

The proof of this observation can be found for example in [2].

If we want to use the CCC topology as a basic interconnection network for the Bluetooth Scatternet formation, we have to be careful and consider the roles for masters and slaves. We propose that nodes in the CCC network are represented by masters in the Scatternet network. Each link from the CCC network will be implemented by a slave (called also a bridge) belonging to two masters and no slave slave will be connected to more than two masters. We can observe that for simulating d-dimensional CCC we need to have at least $5 \cdot d \cdot 2^{d-1}$ nodes ($d \cdot 2^d$ masters and $3 \cdot d \cdot 2^{d-1}$ slaves). It is possible for each master to have 4 additional slaves, thus the upper bound on the number of nodes in d-dimensional network is $13 \cdot d \cdot 2^{d-1}$.

When the number of devices participating in the network exceeds this number, we have to start a process which will rebuild the network. The easiest way would be just to increase d by 1. However, this solution would not work due to the lower bound on the required number of nodes in a $d+1$-dimensional CCC network.

Therefore we introduce an intermediate network topology between the d-dimensional CCC and the $(d+1)$-dimensional CCC. The d-dimensional intermediate CCC network, or in short d-dimensional iCCC network, is defined as follows:

Definition 2. *The d-dimensional iCCC network has $(d+1) \cdot 2^d$ nodes. The nodes are represented by two indices (i,j), where $0 \leq i \leq d$ and $0 \leq j < 2^d$. The connectivity is:*

$$(i,j) \rightarrow \begin{cases} (i, j \oplus 2^i) & 0 \leq i < d, 0 \leq j < 2^d \\ ((i \pm 1) \mod (d+1), j) & 0 \leq i \leq d, 0 \leq j < 2^d \end{cases}$$

The first set of edges are the cube edges; the second set of edges are the cycle edges.

Compared to the standard CCC definition, the iCCC topology contains an additional ring node (d,j) for each ring of the CCC. This additional ring node is connected to the nodes $(d-1, j)$ and $(0, j)$. As node $(d, j + 2^{(d+1)})$ does not exist, node (d, j) does not have a cube edge.

The properties of the iCCC network are very similar to the properties of the CCC network topology:

Observation 2. *The d-dimensional iCCC network has the following properties:*

1. *The number of nodes is* $n = (d+1) \cdot 2^d$.
2. *The degree of each node is* 3 *(or smaller for each ring node* (d, j) *or in case where* $d \leq 2$*).*
3. *The number of edges is* $m = (3 \cdot d + 2) \cdot 2^{d-1}$ *(or smaller for* $d \leq 2$*).*
4. *For any two nodes a and b we can compute a path from a to b of length at most* $4 \cdot d$.

The properties 1 to 3 directly follow from the definition of the d-dimensional iCCC network. Observation 4 can be derived from the properties of a d-dimensional CCC network.

To get from a node (i, j) to a node (u, v), the following path selection strategy can be used. The first part of the path is to get from node (i, j) to node $(0, j)$, which takes at most $d/2$ steps. Then a standard routing scheme for the CCC network, which does not consider iCCC specific nodes (d, j), can be used. To achieve this, almost any dimension-order routing scheme can be used, involving not more than $3 \cdot d$ steps. The last part of the path selection, incurring at most than $d/2$ steps, is to get from node (x, v) to node (u, v). This finishes the proof of Observation 2.

For ease of explanation, we assume that the *CCC* and *iCCC* networks have got a dimension of at least 3. Similar to the lower and upper bounds for Scatternets using the CCC as network topology, the upper and lower bounds for an *iCCC* network are as follows:

$$min_d^{iCCC} = (5 \cdot d + 4) \cdot 2^{d-1} \qquad max_d^{iCCC} = (13 \cdot d + 14) \cdot 2^{d-1}$$

A Smooth Way to Maintain the CCC Topology. Below we introduce a maintenance scheme that will involve a smooth transition from a d-dimensional CCC network over a d-dimensional iCCC network to a $(d+1)$-dimensional CCC topology or vice versa. The different steps of this scheme are displayed in Fig. 1. During the transition, for some time each master will have to simulate the behavior of two nodes in the CCC network. Therefore the degree of a master can grow up to 6. This does not cause any problems, since the Bluetooth specification allows a degree of 7.

The transition from a d dimensional network to a $d+1$ dimensional network involves several steps:

At first we only extend each cycle by an additional master numbered d and transform the network into an iCCC network. Therefore, each time if a new node enters the system and cannot become a loose slave, it extends one of the rings by an additional master d (see Fig. 1.b). To connect to the master nodes 0 and $d-1$, two bridge nodes are required. As the first bridge we can use the slave

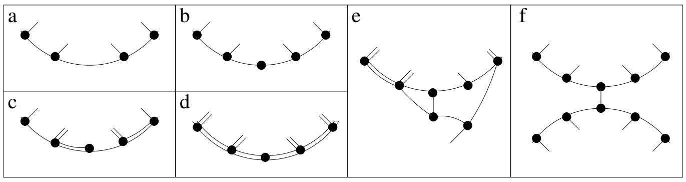

Fig. 1. Transition from a d-dimensional CCC to a $d+1$-dimensional CCC

node that has formerly connected the nodes 0 and $d-1$. As the second bridge we have to take one of the slave nodes of this ring. This transition can be done locally inside each ring. After this step has been made for all rings, the transition to a d-dimensional iCCC is finished.

After the length of each ring has been increased by one, each master acts as two nodes of the $d+1$-dimensional CCC but still has degree 3. From now on, each master wants all of its connections to be doubled. This can also be done gradually as new nodes come and join the network as loose slaves (see Fig. 1.c and 1.d). When a master has doubled all of its connections, it wants to split itself into two nodes, each of them taking over one of the connections from each pair. At this point we distinguish between two types of masters.

A master (d, j) splits itself as soon as it has two loose slaves and both of its edges are doubled. One of its slaves becomes master $(d, j+2^d)$ and the other becomes a bridge between (d, j) and $(d, j+2^d)$. Both pairs of cycle edges are treated in the same way. We describe the procedure for the edges which were both originally connected to $(0, j)$. If the node $(0, j)$ has not split yet, we simply use the edges to connect (d, j) to $(0, j)$ and $(d, j+2^d)$ to $(0, j)$. If it has, we connect (d, j) to $(0, j)$ and $(d, j+2^d)$ to $(0, j+2^d)$ (see Fig. 1.e).

For $i \neq d$, a master (i, j) splits itself as soon as it has a loose slave and all three of its edges (two cycle edges and one cube edge) are doubled. It uses the slave to create master $(i, j+2^d)$ (there will be no connection between (i, j) and $(i, j+2^d)$). One edge from each pair of edges stays connected to (i, j) and the other is connected to $(i, j+2^d)$. To decide which edge is connected to which master, we use the same procedure as for master (d, j). If a node on the other side of the edges has not yet split, we do it arbitrarily. If it has, we do it so that we achieve the following connections: (i, j) with $(i, j \oplus 2^i)$, $((i+1) \mod d, j)$, $((i-1) \mod d, j)$; and $(i, j+2^d)$ with $(i, j \oplus 2^i + 2^d)$, $((i+1) \mod d, j+2^d)$, $((i-1) \mod d, j+2^d)$ (see Fig. 1.e and 1.f).

After all the masters have split, we increase the dimension d by 1.

If a node wants to leave the network, our algorithm works in general inversely to the situation when a node joins the network. The main assumption is that we can exchange the leaving node with any other node in the network. Thus, we can decide which node actually leaves.

Reduction of the network proceeds in three phases. If there are any loose slaves, they are removed in the first place. If there are none, we try to find such $0 \le i \le d$ and $0 \le j < 2^{d-1}$ that node $(i, j+2^{d-1})$ still exists and is independent from node (i, j). We remove the node $(i, j+2^{d-1})$ and attach all of its slaves to the node (i, j). It will now perform the roles of both these nodes. We were allowed to attach all the slaves from one node to the other due to the fact that there were no loose slaves at any of those nodes, so they both had at most 3 slaves each.

If we cannot find either loose slaves or independent masters numbered $(i, j + 2^{d-1})$, we remove double connections, i.e. if we are able to find a pair of masters, that have two bridges between them, we remove one of the bridges. Last of all we can remove nodes $(d-1, j)$ for $0 \leq j < 2^d - 1$ one by one, finally decreasing the dimension from d to $d-1$. When we remove such a node, we use one of the slaves that connected it to $(d-2, j)$ and $(0, j)$ to connect $(d-2, j)$ and $(0, j)$ and the other slave can become a loose one of $(0, j)$.

At the same time as removing the double edges, i.e. after all the masters have been merged in pairs, we decrease the dimension d by 1.

4 Comparison of the Maintenance Scheme with the Best Possible Strategy

If a node enters or leaves the network, the topology of the network changes. Each change of the topology causes costs in terms of interruptig the current communication traffic. To compare our strategy with the best possible strategy, we introduce the following, simple cost model:

Definition 3. *Each insertion or removal of a connection costs one cost unit.*

In the following theorem, we assume that the best possible strategy has only to change one connection for each insertion or removal of a node.

It is possible to show that even in this cost model the additional costs induced by our smooth strategy compared to the best possible strategy can be bounded by a constant factor.

Theorem 3. *The smooth maintenance scheme for the CCC scatternet construction is 6-competitive for the insertion and 20-competitive for the removal of nodes compared with a best possible strategy.*

The proof is available in the full version of the paper.

References

[1] Bluetooth Special Interest Group: Bluetooth SIG, Specification of the Bluetooth System, Ver. 1.2. (2003)

[2] Leighton, F.T.: Introduction to parallel algorithms and architectures: array, trees, hypercubes. Morgan Kaufmann Publishers (1992)

[3] Stojmenovic, I.: Dominating set based bluetooth scatternet formation with localized maintenance. In: Proc. of the 16th IEEE International Parallel and Distributed Processing Symposium (IPDPS). (2002)

[4] Zaruba, G., Basagni, S., Chlamtac, I.: Bluetrees – scatternet formation to enable bluetooth based ad hoc networks. In: Proc. of the IEEE International Conference on Communications (ICC). (2001)

[5] Petrioli, C., Basagni, S., Chlamtac, I.: Bluemesh: degree-constrained multi-hop scatternet formation for bluetooth networks. Volume 9. (2004) 33–47

[6] Petrioli, C., Basagni, S., Chlamtac, M.: Configuring bluestars: multihop scatternet formation for bluetooth networks. IEEE Transactions on Computers, Special issue on Wireless Internet (2003) 779–790

[7] Salonidis, T., Bhagwat, P., Tassiulas, L., LaMaire, R.: Distributed topology construction of bluetooth personal area networks. In: Proc. of the 20th IEEE Infocom. Volume 3. (2001) 1577–1586

[8] Ramachandran, L., Kapoor, M., Sarkar, A., Aggarwal, A.: Clustering algorithms for wireless ad hoc networks. In: Proc. of the 4th international workshop on Discrete algorithms and methods for mobile computing and communications. (2000) 54–63

[9] Baatz, S., Bieschke, C., Frank, M., Khl, C., Martini, P., Scholz, C.: Building efficient bluetooth scatternet topologies from 1-factors. In: Proc. of the IASTED International Conference on Wireless and Optical Communications (WOC). (2002) 300–305

[10] Tan, G., Miu, A., Guttag, J., Balakrishnan, H.: An efficient scatternet formation algorithm for dynamic environments. In: Proc. of the IASTED Communications and Computer Networks (CNN). (2002) 68–74

[11] Wang, Z., Thomas, R., Haas, Z.: Bluenet – a new scatternet formation scheme. In: Proc. of the 35th Annual Hawaii International Conference on System Sciences (HICSS). Volume 2. (2002) 61–69

[12] Lin, T.Y., Tseng, Y.C., Chang, K.M.: A new bluering scatternet topology for bluetooth with its formation, routing, and maintenance protocols. Wireless Communications and Mobile Computing **3** (2003) 517–537

[13] Chang, C.T., Chang, C.Y., Sheu, J.P.: Constructing a hypercube parallel computing and communication environment over bluetooth radio system. In: Proc. of the IEEE International Conference on Parallel Processing. (2003) 447–454

[14] Barriere, L., Fraigniaud, P., Narayanan, L., Opatrny, J.: Dynamic construction of bluetooth scatternets of fixed degree and low diameter. In: Proc. of the fourteenth ACM-SIAM Symp. on Discrete Algorithms (SODA). (2003) 781–790