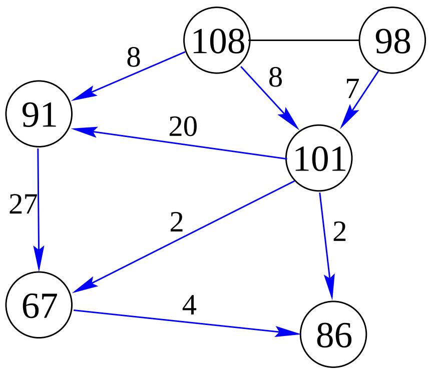
Spectral Methods for Efficient Load Balancing Strategies



Dissertation

von

Robert Elsässer

Schriftliche Arbeit zur Erlangung des Grades
eines Doktors der Naturwissenschaften

Fachbereich Mathematik/Informatik
Universität Paderborn

Paderborn, 25.02.2002

DANKSAGUNGEN

Für die umfangreiche Betreuung meiner Arbeit durch viele Anregungen und interessante Diskussionen bedanke ich mich bei Prof. Dr. Burkhard Monien. Die enge Zusammenarbeit mit ihm hat meine wissenschaftliche Arbeit in den letzten Jahren maßgeblich geprägt. Prof. Dr. Sergej Bezrukov danke ich für die hervorragende Betreuung während meines Studiums. Ein weiterer Dank gilt Prof. Dr. Friedlhelm Meyer auf der Heide und Prof. Dr. Andreas Frommer für die fruchtbare Zusammenarbeit und für die Begutachtung dieser Dissertation.

Weiterhin bedanke ich mich bei den Kollegen aus der Arbeitsgruppe Monien. Insbesondere bei denjenigen, mit denen ich eng wissenschaftlich zusammengearbeitet habe: Rastislav Královič, Thomas Lücking, Dr. Robert Preis, Stefan Schamberger und Dr. Ulf-Peter Schroeder.

Neben den schon genannten Personen existiert durch den Sonderforschungsbereich, das Heinz Nixdorf Institut und das PC^2 ein fruchtbares wissenschaftliches Umfeld in Paderborn, für das ich mich bei allen Mitgliedern bedanke.

Ein Dankeschön an Eric Torunski für die sprachliche Aufbereitung dieser Dissertation.

Ich möchte mich bei meinen Eltern bedanken, die mich immer und mit all ihren Kräften unterstützt haben.

Nicht zuletzt bedanke ich mich bei Brigitta Várkondi für die Unterstützung und Aufmerksamkeit, die sie mir zu jeder Zeit und in jeder Hinsicht zu schenken wusste.

Robert Elsässer

Diese Dissertation wurde von den folgenden Projekten unterstützt.

- Deutsche Forschungsgemeinschaft
  Sonderforschungsbereich 376 'Massive Parallelität'

- Europäische Union ESPRIT - Programm
  Long Term Research Project 20244 'ALCOM-IT'

CONTENTS

1. INTRODUCTION

The problem of balancing dynamically generated work load among the processors of a parallel machine occurs in a wide range of applications. In Section 1.1, we consider examples of such applications and describe the importance of efficient load balancing strategies. In Section 1.2, we present the most common approaches used to solve the load balancing problem and give a short overview concerning our main results. In Section 1.3, we briefly introduce some related load distribution problems and in Section 1.4, we conclude with an outline of the thesis.

1.1 Motivation

Parallel processor machines are widely used in all areas of science and technology. In order to run a process on a parallel machine, it must be possible to divide the process into subprocesses. Each subprocess has a computational load, and some interdependencies between these subprocesses may exist. Therefore, the process can be modelled by a *process graph*, in which the vertices correspond to the subprocesses and the edges correspond to the interdependencies.

To minimize the overall computation time, the total work load has to be distributed evenly among all processors of the parallel machine. Additionally, the amount of communication between the processors should also be reduced. Hence, the process graph has to be partitioned into k equal parts with k being the number of processors and, as a second criterion, the number of edges between vertices of different parts should be minimized. In other words, the so-called *k-partitioning problem* has to be solved on the process graph.

Examples showing the importance of efficient load balancing methods are parallel finite element simulations. They appear in computational fluid dynamics, crash simulations, weather forecasts or earthquake simulations. In all these applications, *partial differential equations* (PDEs) are used to describe the behavior of some physical systems. Since analytical solutions of such problems exist only in very restricted cases, the continuous space, defined by the PDE, is transformed into a discrete space. Mainly, simple objects, such as triangles or rectangles, are used for this transformation, obtaining a mesh with several millions of elements. The solution of the PDE is then approximated by solving a system of linear equations described by this *finite element mesh*. For a detailed description, please refer to [BJ93, Zie89]. See also Figure 1.1 (left), where a finite element mesh is shown.

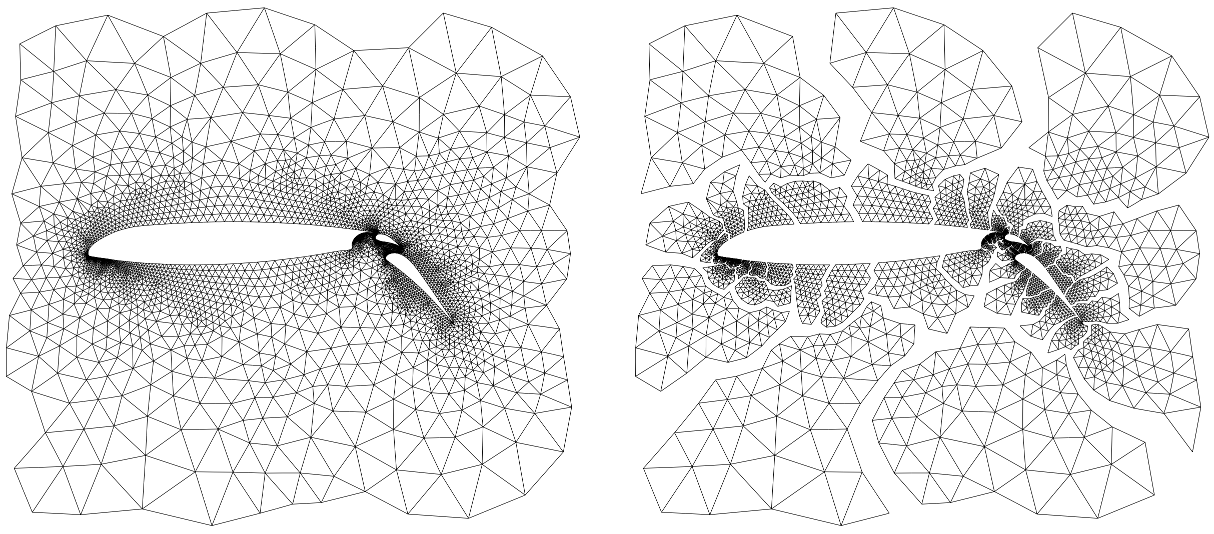


Fig. 1.1: Partitioning a graph obtained from a finite element simulation into 64 parts

Since the computational demands of these applications can not be met by a single processor machine, parallel algorithms must be used to solve the system of equations described above. Therefore, the mesh is partitioned into several subdomains and each subdomain is then assigned to a different processor of the machine. In order to not affect the speed of computations, the assignment should be uniform, and the data exchange between the processors should be minimized.

To solve this partitioning problem, most common approaches (see e.g. [Die98]) construct a graph, such that the vertices represent the elements of the mesh (in our case the triangles or rectangles), and an edge connects two vertices if the corresponding elements have a common side. Now, the goal is to partition the vertices of this graph into k equal-sized subsets so that the number of cut edges, separating the subsets from each other, is minimized. This minimal number of cut edges of a balanced optimal k-partition is called the *k-section width* of the graph. Figure 1.1 (right) illustrates the partitioning of the finite element mesh, shown on the left, into 64 parts.

Depending on the application, the mesh refines and coarsens in some areas during the computation, which causes an imbalance between the processors' load, and therefore delays the overall computation. In fluid dynamics for example, simulation of turbulences or shocks often depends on such refinements. In particular, in regions with steep solution gradients, the mesh has to be sufficiently refined. In these situations, the load should be rebalanced. Therefore, the application is interrupted and the current static load balancing problem is solved.

One approach to solve the load balancing problem is based on repartitioning. In this

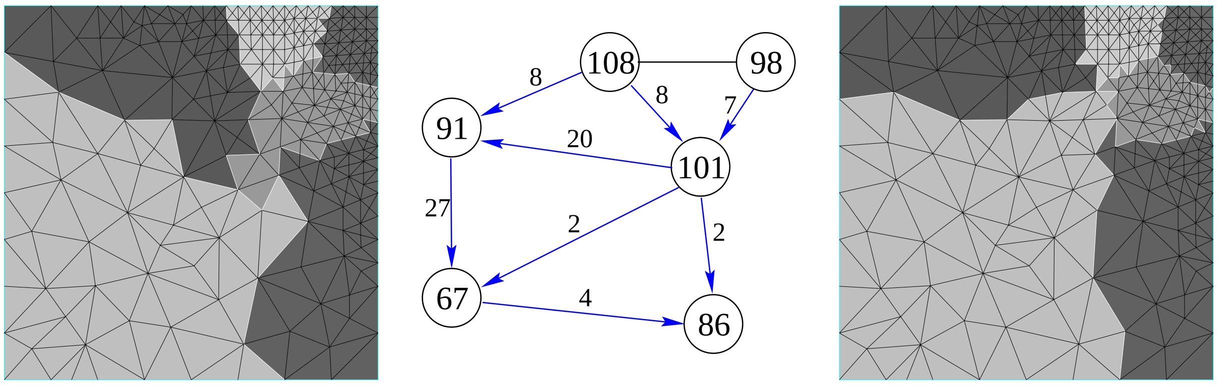


Fig. 1.2: An unbalanced partition of a mesh (left), the corresponding quotient graph with the balancing flow (center), and the obtained balanced partition (right).

case, a completely new partitioning is computed from scratch. In most cases, this method yields a much smaller cut size than approaches based on local improvements. The drawback of this method is the large amount of data that has to be transferred between the processors. Techniques based on repartitioning are analyzed in e.g. [OB98], where attempts to reduce the communication complexity are also presented.

We deal with load balancing algorithms, which start from the existing partitioning and act in three phases. The first phase answers the question *"how much"* load has to be shifted over the edges of the so-called *quotient graph*, i.e. determines the *balancing flow*. The quotient graph is defined by the adjacencies between subdomains and contains a node for each subdomain. Edges represent common borders between the subdomains. Figure 1.2 shows an unbalanced partition of a simple mesh into six subdomains (left), the resulting quotient graph (center) and a balanced partitioning of the mesh (right).

The problem of determining the balancing flow can be modelled by a system of linear equations [HBE98]. To solve the system of equations, we use iterative algorithms performing locally on the quotient graph, i.e. the nodes of the graph exchange load information with their direct neighbors in iterations until a balancing flow is computed. Related to this context, see also [Fro90] w.r.t. algorithms solving systems of linear equations on parallel machines.

Once the balancing flow is determined, the second phase computes the scheduling of the flow. If the current load of each processor is higher than the total load that has to be sent to its neighbors, then the processors can balance their loads in one step and no scheduling is needed. However, in most cases this condition does not hold and therefore, we use a scheduling phase to decide in which order the load has to be moved.

In the third phase, some elements are chosen and the load is sent according to the schedule. The choice of the elements may consider additional criteria, such as minimizing the cut size or optimizing the shape of subdomains. For the last one, see e.g. [DMM98, FMB95, JAMS89, VFC$^+$96, WCDS99].

Other examples showing the importance of efficient load balancing strategies are applications where jobs without communication dependencies are somehow placed on the processors of a parallel machine. In order to avoid idle times, we should redistribute the jobs among the nodes of the network in such a way that we obtain a balanced load situation. If the topology of the parallel system is described by a complete graph, then we can solve the load balancing problem in one step. Otherwise, we use the load balancing method acting in three phases as described above. To calculate the balancing flow, we propose local iterative algorithms performing on the processor graph. Other efficient load balancing strategies for such applications are briefly described in section 1.3.

1.2 Problems and Solutions

This section is divided into two subsections. In Subsection 1.2.1, we describe the most common approaches applied to solve the k-partitioning problem and present the techniques that are used to analyze the quality of k-sections in graphs. We also describe our main results concerning the k-section width of graphs.

In Subsection 1.2.2, we specify different local iterative load balancing algorithms and analyze their behavior. We show that the convergence rate of the most common schemes, executed on a network, depend on some eigenvalues of the corresponding topology. We conclude with an outline of our results concerning iterative load balancing algorithms.

1.2.1 The k-Partitioning Problem

It is easy to find optimal k-sections only for "trivial" graphs as e.g. cycles, paths and complete graphs. The calculation of the k-section width for arbitrary graphs is *NP*-complete. This is already true for the bisection problem [GJS76], and it remains *NP*-complete by considering only d-regular graphs [BCLS87]. However, finding the k-section width is very difficult even for graphs, for which computing the bisection width is easy, such as for hypercubes or grids [Bez96, BR97].

Many papers in the literature deal with the design of partition algorithms [DLMS96, GMT95, HM92, MD97, DMP95] and the analysis of their optimality [CY94, GM95], or with the complexity of partition problems. Different methods of calculating partitions of a graph have been proposed in the past by scientists of different fields like mathematics, computer science or engineering. There are also many software libraries available, which include the most efficient graph partitioning approaches [FLS95, Gup96, HL94, PD97, Pel96, Wal00].

Some applications use recursive bisection to obtain k-sections of graphs. In order to compute upper bounds on the bisection width, global and local methods have been developed. The most common global methods are *inertial-, spectral-* and *geometric-partitioning* [DMP95, GGL93, Ham92, HL95], while efficient local methods consist of the so-called *Kernighan-Lin* [KL70] and the *helpful-set heuristic* [DMP95, HM92, MD97, MP01].

For the evaluation of the quality of partition algorithms, it is helpful to know good lower bounds for the k-section width. They can also be used to speed up Branch & Bound strategies, which calculate k-sections for graphs with a moderate number of vertices. Furthermore, by considering the case $k = 2$, there is a demand to construct graphs with a high bisection width in order to be used as a topology for routing networks. In this context, lower bounds guarantee a minimum communication bandwidth between any two balanced parts of the network.

There are only a few known approaches, which compute lower bounds for the k-section width of a graph. Leighton proposes a lower bound of the bisection width by calculating a routing scheme between all pairs of vertices such that the congestion is minimized [Lei92]. This technique can also be extended for the k-section width.

Lower bounds on the size of optimal k-partitions can also be derived from algebraic graph theory by relating the k-section problem to an eigenvalue problem. The classical lower bound on the k-section width ∇ of a graph $G = (V, E)$ with $n = |V|$ is

$$\nabla \geq \sum_{i=1}^{k} \lambda_i \cdot \frac{n}{2k}, \tag{1.1}$$

where $\lambda_1, \ldots, \lambda_k$ are the k smallest eigenvalues of the Laplacian of G (see Chapter 2 for the definition of the Laplacian of a graph) [DH73].

In [FRW94], Rendl and Wolkowicz get lower bounds using a different approach. They define the k-section problem as a semidefinite program and obtain the well-known lower bound (1.1) stated above. Furthermore, they apply projection techniques from continuous optimization [HRW92] to derive better results [RW95].

In [Bol93], Bolla considers the spectra of hypergraphs and derives spectral bounds on their k-partitioning size. In [BT94], Bolla and Tusnády also use spectral techniques to handle combinatorial problems concerning minimal k-cuts of weighted graphs. However, they do not require balanced partitioning sets.

In Chapter 3, we show that in (1.1) equality holds for a restricted number of graphs. Nevertheless, there is generally a large gap between this lower bound and the k-section width. To give an example, the mesh used to discretize the object of crash or flow simulations often has a structure similar to a grid. The first three smallest nonzero eigenvalues of a $\sqrt{n} \times \sqrt{n}$ torus are $\lambda_2 = \lambda_3 = \lambda_4 = 2 - 2\cos(\frac{2\pi}{\sqrt{n}}) \approx 4\pi^2/n$. Obviously, the 4-section width is $4\sqrt{n}$. However, the lower bound of inequality (1.1) does only result in a value of

$\frac{3}{2}\pi^2$. So there is a quadratic gap between the classical lower bound and the real value of the sum of the first k eigenvalues for a small k. Deriving new lower bounds, which use $\sqrt{\sum_{i=1}^k \lambda_i}$ instead of $\sum_{i=1}^k \lambda_i$ for graphs with a grid-like structure, we close this quadratic gap up to a constant factor for this type of graphs. Note that according to Spielman and Teng, $\lambda_2 = O(\frac{1}{n})$ holds for bounded-degree planar graphs and two-dimensional meshes. Furthermore, $\lambda_2 = O(\frac{1}{n^{2/d}})$ for well-shaped d-dimensional meshes [ST96].

Such a quadratic gap also appears in the relation between the edge-expansion and λ_2 [Che70, DK86, Moh89]. The edge-expansion of a graph $G = (V, E)$ is defined by $i(G) = \min_{S \subset V} \frac{|E(S,\overline{S})|}{\min\{|S|,|\overline{S}|\}}$, where $E(S, \overline{S})$ is the set of edges connecting vertices of S with vertices of $\overline{S}$. In [DK86, Moh89], it is shown that

$$\frac{i^2(G)}{2 \cdot d_{max}(G)} \leq \lambda_2 \leq 2 \cdot i(G),$$

where $d_{max}(G)$ is the maximal vertex degree of G.

To determine a new upper bound for $\sum_{i=1}^k \lambda_i$, we make use of the level structure of a k-section, in which every level consist of all vertices having the same distance to the cut. This leads to a generalized lower bound of (1.1) depending on the growth of the sizes of these levels in each section. Let $g : I\!N \to I\!N$ be a function. We will introduce the class of Level Structured graphs $LS(g, \nabla, k)$, which have a k-section with a cut size of ∇ and a level structure such that there are no more than $\nabla g(j)$ edges connecting vertices of distance j to the cut with vertices of distance $j+1$ to the cut. We derive improved relations between the k-section width and $\sum_{i=1}^k \lambda_i$ for the graph class $LS(g, \nabla, k)$. If the sum $\sum_{j=2}^{\infty} \frac{1}{g(j-1)}$ is bounded, then these new relations differ from the classical bound (1.1) in a constant factor. In other cases, we prove that for $\frac{n}{\nabla} \to \infty$, there exists a constant δ such that

$$\nabla \geq \delta \left(\sum_{i=1}^k \lambda_i \right)^{\beta} n(1 - o(1)), \tag{1.2}$$

where β is in the range $\frac{1}{2} \leq \beta < 1$ and n is the cardinality of the graph. We also show that there are graphs, for which the new bounds are tight up to a constant factor. These new bounds may also be used the other way around to get an upper bound of $\sum_{i=1}^k \lambda_i$ from any k-section with a cut size ∇ and a growth function g.

In general, the level structure of an optimal k-section is not known, but there are notable exceptions. If G is a graph of maximum degree d, then $G \in LS(g, \nabla, k)$ with $g(j) = (d-1)^j$. Thus, for $\frac{n}{\nabla} \to \infty$, we obtain the following inequality

$$\nabla \geq \frac{k(d-2)+2}{k(d-2)} \cdot \frac{\sum_{i=1}^k \lambda_i \cdot n}{2k}(1 - o(1)).$$

We expect that there exist further classes of graphs, for which we can directly derive better lower bounds on the k-section width using the results of this thesis.

1.2.2 Local Iterative Load Balancing

As described in Section 1.1, one of the most common approaches for local load balancing is the 3-phase model (e.g. [DFM99]). We focus on the first phase and analyze the questions: how much load has to be migrated and where to? Formally, given a network with n nodes, where node i contains work load w_i, we calculate a balancing flow over the edges of the network such that, after the third phase, each node i has the balanced work load of $\overline{w_i} = \sum_{j=1}^{n} w_j/n$. We further assume that no load is generated or consumed during the balancing process and the structure of the network is fixed, i.e. we consider a static load balancing scenario.

We described in Section 1.1, that it is possible to model the load balancing problem by a system of linear equations [HBE98]. Several methods exist, which solve the problem by computing first the global imbalance vector $w - \overline{w}$ (see e.g. [Die98, HB95])

Assuming that processors of the parallel network may only access information of their direct neighbors, we consider algorithms, which exchange load information locally, in iterations, until a balancing flow is computed. Two subclasses of local iterative load balancing algorithms are *diffusion schemes* [Boi90, Cyb89] and *dimension exchange schemes* [Arn01, Arn02, Cyb89, XL97]. These two classes reflect different communication abilities of the network. Diffusion algorithms assume that a processor can send and receive messages to/from all of its neighbors simultaneously, while the dimension exchange approach is more restrictive and allows a processor to communicate with one of its neighbors during each iteration.

In [Cyb89], Cybenco defined the general diffusion scheme. If we denote with w_i^k the load after the k^{th} iteration step on node i of the graph $G = (V, E)$, then w_i^k satisfies the equation

$$w_i^k = w_i^{k-1} - \sum_{\{i,j\} \in E} \alpha_{i,j}(w_i^{k-1} - w_j^{k-1}).$$

Most of the results in this area concentrate on homogeneous schemes with $\alpha_{i,j}$ being the same for any $\{i, j\} \in E$.

There is plenty of work (e.g. [Cyb89, DFM99, DSW98, GMS96, SKK97, WCE97]) focusing on the relation between convergence rates of diffusion algorithms and the condition number of the unweighted Laplacian (see Chapter 2 for the definition). It is known that by increasing the condition number, the convergence rate of most common diffusion schemes also increases. In [DFM99], it is proved that all diffusion schemes calculate the same flow and that this flow is minimal w.r.t. the l_2-norm. In the same paper, it is also shown that the known diffusion schemes can be generalized for edge-weighted graphs. Sending a higher amount of load over heavier weighted edges, the calculated flow is still minimal with respect to a weighted l_2-norm. A formal definition is given in the next chapter.

So far, little work has been done to address load balancing algorithms for heterogeneous networks. Considering inhomogeneous schemes described by edge-weighted graphs, the objective is to find edge-weights such that the condition number of the resulting Laplacian is maximized among all Laplacians having the same communication structure, e. g. having the same zero entries. At this time, very little is known about this problem. To our knowledge, [DMN97] was the first and, up to now, the only paper addressing this topic. There, semidefinite programming is used and it is proved that a polynomial time approximation algorithm exists, which approximates the optimal values. Furthermore, some examples of graph classes with optimal weights are given. Using this approach, however, noticeable results can only be obtained for graphs of small cardinality.

In Chapter 5, we consider edge-transitive graphs and show that for these graphs the condition number is maximized if all edges have the same weight. This result solves some open problems described in [DMN97] with respect to optimal edge-weights of hypercubes, cycles, and the star. Next, we consider Cayley graphs and prove that edges generated by the same generator must be of equal weight in order to maximize the condition number. Another general graph class consists of Cartesian products of graphs. For this class, we compute edge-weights that can be used to improve known load balancing diffusion algorithms on graphs belonging to this class. Additionally, we analyze Cube Connected Cycles and related hypercubic networks. We compute optimal values for the weights of their edges, maximizing the condition number of the corresponding Laplacian. To confirm the theoretical results, we describe several experiments with different edge-weight scenarios on the previously mentioned graph types and show some dependencies between edge-weights and convergence rate.

Heterogeneous networks, consisting of processors of different computing power or memory capacity, can be modelled by node-weighted graphs. These networks are extremely attractive because they often appear as computer networks containing processors from different manufacturers and of different types. Consult [SG99] for an overview of the evolution of heterogeneous concurrent computing in the context of the parallel virtual machine (PVM).

In Chapter 6, we consider the load balancing problem in heterogeneous processor networks. Since the processors of a network have different work loads and different processor speeds, this can result in unequal remaining processing times. Therefore, the load has to be balanced among the processors proportional to their computing power. Formally, given a network with n nodes, where each node i has work load w_i and weight c_i, calculate a load balancing flow over the edges of the network such that, after the load balancing process, node i has the balanced proportional work load of

$$\overline{w_i} := \frac{\sum_{j=1}^{n} w_j}{\sum_{j=1}^{n} c_j} c_i \ .$$

Then, the time to consume the load on each node is equal. Obviously, this problem generalizes the load balancing problem on homogeneous networks, where $c_i = 1$ for all

$i \in \{1, \ldots, n\}$. We assume that the situation is fixed, i.e. no load is generated or consumed during the balancing process and the structure of the network does not change.

In Chapter 4, a combination of diffusion and dimension-exchange is discussed, which is sometimes called the *Alternating Direction Iteration* (ADI) scheme [Var62]. It can be used if the network is the Cartesian product of two graphs. Examples are $k \times k$ square tori (Cartesian product of two cycles of k vertices each), $k \times k$ square grids (Cartesian product of two paths of k vertices each) or a hypercube of dimension k (Cartesian power with dimension k of a path of 2 vertices), which often occur as processor networks. In this model, in every iteration a processor first communicates with its neighbors along one component of the product and then with its neighbors along the other component of the product. We will show that for graphs satisfying the Cartesian property, the number of load balancing steps can be reduced to the half if the load balance process alternates between balancing along one component and the other. As drawback of this scheme, we can prove that the resulting flow may be very large if optimal parameters for the number of iterations are used. To avoid this problem, we present the *Mixed Direction Iterative* (MDI) scheme, which needs the same number of iterations, but results often in a much smaller flow.

In [DFM99], a new optimal load balancing scheme was introduced, which is based on the spectrum of the graph. Only $m-1$ balancing steps are needed with m being the number of different eigenvalues of the graph. The calculated flow is minimal according to the l_2-norm. It is also specially constructed to keep the load-differences during the calculation as small as possible, i.e. it is very stable from the numerical point of view. In [EFMP99], a much simpler optimal scheme OPT is presented, which can easily be applied to several specific graphs with a known spectrum. It also needs $m-1$ iterations to balance the load. Although this scheme might get trapped in numerically instable conditions, there are also rules on how to avoid them.

In order to show the power of such an optimal scheme, consider a process containing subprocesses without any communication dependencies and a complete graph as the processor graph. This is the case in a bus system, where each processor can communicate with any other processor in the network. In order to avoid high communication costs, we allow any processor to communicate only with a small number of other nodes in the system. In this sense, we define a topology, which has a small vertex degree and a small number of different eigenvalues. Applying the OPT scheme, the maximum degree and the number of different eigenvalues give an upper bound for the number of steps needed to calculate a balancing flow. Various sparse network topologies having a small spectrum are compared and proposed in [EKM01]. See also [DMP00] for a practical point of view on this problem.

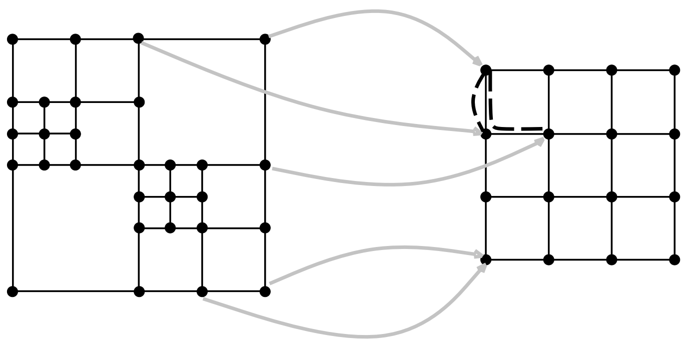


Fig. 1.3: Embedding a process graph (left) into a processor graph (right). Edges of the process graph are mapped onto routing paths in the processor network.

1.3 Related Problems

In the previous section, we described the problem of distributing the vertices of a process graph among the nodes of a processor network, by minimizing the communication between the processors. If any pair of two adjacent vertices in the process graph can be mapped to the same processor or to two adjacent processors of the processor graph, then the cut size of the partitioning represents the communication complexity between the processors. Otherwise, the so-called *graph embedding problem* has to be also taken into consideration (see e.g. [Sch00]).

An embedding of a guest graph $G = (V, E)$ into a host graph $H = (V', E')$ is a function $\phi : V \to V'$ together with a routing scheme, which assigns to each edge $\{u, v\} \in E$ a path from $\phi(u)$ to $\phi(v)$. We denote with *dilation of an embedding* the maximum length of the routing path between any two vertices in H, and with *congestion of an embedding* the maximum number of routing paths along any edge of H. The goal is to find an embedding of G into H, such that the vertices of G are evenly distributed among the vertices of H, and the congestion and/or dilation of the embedding is minimized. See Figure 1.3 for an example.

The graph embedding problem is NP-complete for general graphs [GJ79]. Some optimal embedding functions concerning the dilation and congestion have been developed for pairs of graphs like grids, trees and hypercubes [Lei92, MS90, Röt98, Sch00]. There have also been several heuristics designed to solve this problem. See [dBB$^+$97] for an overview concerning such algorithms.

Throughout the thesis we assume that in a processor system synchronicity is given, i.e. the nodes of the network are synchronized by a global clock. Additionally, we assume that the network does not change its topology during the computations. In the past, some papers have also been published concerning the load balancing problem in asynchronous and dynamic networks. In [AAMR93, GLM$^+$95], the authors analyzed an algorithm for the token distribution problem (see also [PU87, PU89, MOW96]) in synchronous and static, dynamic, and asynchronous networks. They stated a relation between the number of balancing steps of this algorithm and the edge- or node-expansion of the processor network. However, they did not require a totally balanced situation to stop the algorithm. Other articles concentrated on the question if known iterative algorithms also converge in dynamic and asynchronous systems (e.g. [BT89, Bah00, SB96, FS00]).

Applying spectral methods, we also assume that the load balancing problem is static, i.e. load is not generated or consumed during the load balancing process. To solve the dynamic load balancing problem several methods have been proposed. One of them consists of the *dynamic mapping*, which places dynamically generated processes onto processors and, once placed, a process has to stay on "its" processor until termination. A number of theoretical results w.r.t. this approach are based on the so called "balls-into-bins" game. In this game, a number of balls have to be placed into a number of bins. It is known that if n balls are placed into n bins at random, then the fullest receives $O(\log(n)/\log(\log(n)))$ balls with high probability. Azar et al. showed that if for each placement $k \geq 2$ bins are asked for their load and the ball is placed into the lightest, then the fullest bin contains only $\frac{\log(\log(n))}{\log(k)} + O(1)$ balls (with high probability) [ABKU94]. Several papers consider algorithms for the parallel placement of balls and discuss extensions to the placement of weighted balls [ACMR95, BMS97, DDLM95, Ber00].

If there are strong communication dependencies between the processes, and additionally, they can not be moved, then the so-called *dynamic embedding problem* occurs. In this case, the most common balancing strategies place dynamically generated jobs onto processors in the neighborhood of their origin. For analytical results and heuristics on some known network topologies concerning this problem, the reader is referred to [HM96, LNRS92, MFKL93, OD93, Ran91].

A large number of efficient load balancing algorithms are known for applications where migratable jobs without communication dependencies are generated dynamically. The first theoretical results concerning this problem were presented in [RSAU91]. In [LK87] and [LMR91], a gradient model is analyzed. Other papers concentrated on algorithms where the processors balance their load with a fixed set of neighbors if the load difference between them increases above a certain threshold [LM92]. In further models, a processor balances its load with a few randomly chosen other processors if its load changes by at least a certain factor (see e.g. [LM93, BFG01]).

1.4 Outline

In Chapter 2, we give some basic definitions and provide the theoretical background of this thesis. We analyze the relation between the k-section width of graphs and the first k eigenvalues of their Laplacian in Chapter 3. Using these results, we derive new spectral lower bounds on the k-section width of large graphs having a certain structure. In Chapter 4, we design new load balancing algorithms for Cartesian products of graphs, and describe a simple but powerful diffusion scheme. Furthermore, we present families of sparse graphs having a small spectrum. In Chapter 5, we improve some known load balancing schemes for edge-transitive graphs, Cayley graphs and several interconnection topologies. In Chapter 6, we generalize all known diffusion schemes to heterogeneous processor systems, and derive new relations between the second smallest eigenvalue of the Laplacian of a node-weighted graph and its edge-expansion. We conclude the thesis with some discussions and open problems.

1.5 Publications

The author has already published related results to this thesis in the journals *Discrete Applied Mathematics* [BES99b], *Journal of Combinatorial Theory* [BE00] and *Annals of Combinatorics* [BDE00]. Articles concerning some parts of the thesis appeared in the Proceedings of the *Euro-Par Parallel Processing Conference* [EFMP99], the *Workshop on Graph-Theoretic Concepts in Computer Science* [BDE99, BEM$^+$00, BE01], the *Conference on Computing and Combinatorics* [BES99a], the *Symposium on Parallel Algorithms and Architectures* [EMP00, ELM01], the *Symposium on Theoretical Aspects of Computer Science* [EKM01] and the *International Parallel and Distributed Processing Symposium* [EMRS02].

2. BASIC DEFINITIONS AND THEORETICAL BACKGROUND

Let $G = (V, E)$ be an undirected graph with V being the set of vertices and E being the set of edges. If G is an edge-weighted graph, then we denote with $c_{v,w}$ the weight of the edge $\{v, w\} \in E$. Since G is undirected, $c_{v,w} = c_{w,v}$ for any $\{v, w\} \in E$.

Definition 1: *Let $G = (V, E)$ be a graph. The function*

$$\pi \ : \ V \rightarrow \{1, 2, \ldots, k\}$$

represents a k-partition of G that distributes the vertices among k parts V_1, V_2, ..., V_k.

In the sequel, we use the notation $V = V_1 \uplus V_2 \uplus \ldots \uplus V_k$ for a k-partition. In the case of $k = 2$, π is called a *bisection* of G. A *balanced k-partition* of a graph G satisfies the condition $||V_i| - |V_j|| \leq 1$ for any $i, j \in \{1, 2, \ldots, k\}$. In Figure 2.1, a balanced bisection of a graph is shown. An important cost measure of a k-partition is its *cut size.*

Definition 2: *Let $G = (V, E)$ be a graph. Then,*

$$\nabla_\pi = |\{\{v, w\} \in E \mid \pi(v) \neq \pi(w)\}|$$

is the cut size *of π. This can be generalized to*

$$\nabla_\pi = \sum_{\substack{\{v,w\} \in E \\ \pi(v) \neq \pi(w)}} c_{v,w}$$

in the case of edge-weighted graphs.

As already mentioned, the minimal cut size ∇ among all balanced k-partitions of a graph G is called the k-section width of G.

As described in the introduction, the problem of computing the k-section width of a graph is *NP*-complete and this is already true for the bisection problem. Optimal k-sections can be computed only for small graphs with less than 100 vertices. However, most

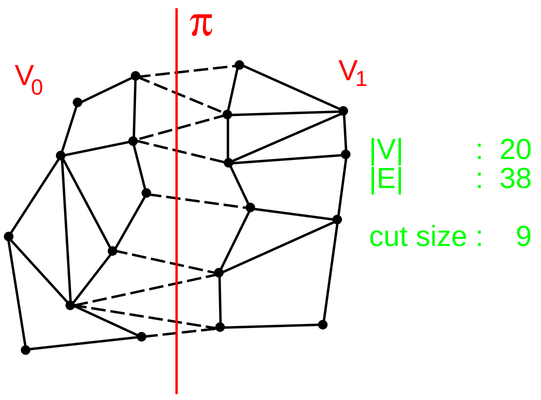


Fig. 2.1: A balanced bisection of a graph

applications are satisfied with a fast calculation of a partition with sufficiently small cut size. The most global and local partitioning heuristics have been developed to perform this task. Some other partitioning tools concentrate on optimizing cost measures such as the shape of the parts or on the minimization of the vertex boundary. See [Pre00] for an excellent overview on different partitioning methods and existing software libraries concerning this field of research.

For a graph $G = (V, E)$ with $|V| = n$, the $n \times n$ *Laplace matrix* $L = \{l_{v,w}\}$ is defined by

$$l_{v,w} = \begin{cases} \deg(v), & \text{if } v = w \\ -1, & \text{if } v \neq w \text{ and } \{v, w\} \in E \\ 0, & \text{otherwise} \end{cases}$$

The Laplacian can be easily generalized for edge-weighted graphs. In this case, $l_{v,w} = -c_{v,w}$ for $\{v, w\} \in E$ and $deg(v)$ represents the weighted degree of v, i.e. $deg(v) = \sum_{\{v,w\}\in E} c_{v,w}$. It is known that all the eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$ of L are non-negative. They have pairwise perpendicular eigenvectors and $(1, 1, 1, ..., 1)^t$ is an eigenvector with eigenvalue 0. For connected graphs, the multiplicity of this eigenvalue equals 1 (see e.g. [PSL90]).

It follows from the Courant-Fisher theorem [Wil65] that

$$\begin{aligned} \lambda_i &= \max_{dim(U)=n-i+1} \left\{ \min_{x\in U} \left\{ \frac{x^t L x}{x^t x} \right\} \right\} \qquad (2.1) \\ &= \max_{dim(U)=n-i+1} \left\{ \min_{x\in U} \left\{ \frac{\sum\limits_{\{u,v\}\in E} (x_u - x_v)^2}{\sum\limits_{v\in V} {x_v}^2} \right\} \right\}. \end{aligned}$$

For the graph $G = (V, E)$ with $|V| = n$ and $|E| = N$ define $A \in \{-1, 0, 1\}^{n\times N}$ to be the *node-edge incidence matrix* of G. A contains a row for each node and a column for each edge. Each column has exactly two non-zero entries "1" and "-1" for the two nodes incident to the corresponding edge. The signs of these non-zeros (implicitly) define directions for the edges of G. In the sequel, these directions will be used to express the direction of the balancing flow. Let $B \in \{0, 1\}^{n\times n}$ be the *adjacency matrix* of G. As G is undirected, B is symmetric. For an unweighted graph G, column/row i of B contains 1's at the positions of all neighbors of v_i. Obviously, $L = AA^T$. We denote with w_i the load assigned to node $v_i \in V$ and let w be the vector of load values. $\overline{w} = \frac{1}{n}(\sum_{i=1}^{n} w_i)(1, \dots, 1)^t$ denotes the corresponding vector of average load. Let $x \in I\!R^N$ be a flow on the edges of G. x is called a *balancing flow* on G iff $Ax = w - \overline{w}$. It expresses the fact that the flow balance at each node corresponds to the difference between its initial load and the mean load value, i.e. after shipping a load of exactly x_e via each edge $e \in E$, the load is globally balanced.

Consider the following *local iterative load balancing algorithm*, which computes a balancing flow x among the edges of G, and is known as the *first order scheme* (FOS) [Cyb89].

$$\begin{array}{rrcl}
\forall\, e = \{v_i, v_j\} \in E: & y_{i,j}^{k-1} & = & \alpha(w_i^{k-1} - w_j^{k-1}); \quad x_e^k \;=\; x_e^{k-1} + \delta_{i,j} y_{i,j}^{k-1}; \\
\text{and} & w_i^k & = & w_i^{k-1} - \sum_{e=\{v_i,v_j\}\in E} y_{i,j}^{k-1}
\end{array} \tag{2.2}$$

Here, we set $\delta_{i,j} = 1$ if $i > j$ and -1 otherwise, $\delta_{i,j} y_{i,j}^k$ is the amount of load sent via edge e in step k, x_e^k is the total load sent via edge e until iteration k and w_i^k is the load of the node i after the k^{th} iteration. This method performs iterations on the nodes of G and require communication with adjacent nodes only. Equation (2.2) can be written in matrix notation as $w^k = Mw^{k-1}$ with $M = I - \alpha L \in I\!R^{n\times n}$. M contains α at position (i, j) for an edge $e = \{v_i, v_j\}$, $1 - \sum_{e=\{v_i,v_j\}\in E} \alpha$ at diagonal entry i, and 0 elsewhere. Then, M has the eigenvalues $\mu_i = 1 - \alpha\lambda_i$ and α has to be chosen such that $1 = \mu_1 > \mu_2 \geq \ldots \geq \mu_n > -1$. Since G is connected, $\mu_1 = 1$ is a simple eigenvalue having $(1, 1, \dots, 1)^t$ as an eigenvector. Such a matrix M is called *diffusion matrix*. We denote with $\gamma = \max\{|\mu_2|, |\mu_n|\} < 1$ the second largest eigenvalue of M according to absolute values and call it the *diffusion norm* of M.

Lemma 1 ([DFM99]): *Let w^0 be an initial load and let $\overline{w} = \frac{1}{n}(\sum_{i=1}^{n} w_i^0)(1, \dots, 1)^t$ be the corresponding average load. Moreover, let $w^0 = \sum_{i=1}^{n} z_i$ be a representation of w^0 in terms of (not necessarily normalized) eigenvectors z_i of M with $Mz_i = \mu_i z_i$, $i = 1, \dots, n$. Then $\overline{w} = z_1$.*

Several modifications to the first order scheme have been discussed in the past. One of them is the *second order scheme* (SOS) [GMS96], which has the form

$$\forall\, e=\{v_i,v_j\}\in E:\ y_{i,j}^{k-1}=\begin{cases}\alpha(w_i^{k-1}-w_j^{k-1}) & ,\quad \text{if } k=1\\ (\beta-1)y_{ij}^{k-2}+\beta\alpha(w_i^{k-1}-w_j^{k-1}), & \text{otherwise}\end{cases}$$

$$x_e^k \;=\; x_e^{k-1}+\delta_{i,j}y_{i,j}^{k-1};\ \text{and}\qquad w_i^k=w_i^{k-1}-\sum_{e=\{v_i,v_j\}\in E} y_{i,j}^{k-1}$$

or in matrix notation

$$w^1=Mw^0 \text{ and } w^k=\beta M w^{k-1}+(1-\beta)w^{k-2},\ k=2,3,\ldots$$

with β being a fixed parameter, whereby fastest convergence is achieved for $\beta=\frac{2}{1+\sqrt{1-\mu_2^2}}$. Again, $\delta_{i,j}=1$ if $i>j$ and -1 otherwise, $\delta_{i,j}y_{i,j}^k$ is the amount of load sent via edge e in step k, x_e^k is the total load sent via edge e until iteration k and w_i^k is the load of the node i after the k^{th} iteration.

The Chebyshev method [DFM99] differs from SOS only by the fact that β depends on k according to

$$\beta_1=1,\ \beta_2=\frac{2}{2-\mu_2^2},\ \beta_k=\frac{4}{4-\mu_2^2\beta_{k-1}},\ k=3,4,\ldots$$

Generalized, any scheme, for which the work load w^k in step k can be expressed in the form $w^k=p_k(M)w^0$, $p_k\in\overline{\Pi}_k$, is called a *polynomial-based load balancing scheme.* Here, $\overline{\Pi}_k$ denotes the set of all polynomials p of degree less than k satisfying the constraint $p(1)=1$. Condition $p_k(1)=1$ implies that all row sums in matrix $p_k(M)$ equal 1. Indeed, defining an algorithmically feasible nearest neighbor scheme, it must be possible to rewrite it as an update process where w^k is computed from w^{k-1} (and maybe some previous iterates) involving *one* multiplication with M only. This means that the polynomials p_k have to satisfy some kind of short recurrence relation.

The convergence of a polynomial-based scheme depends on whether (and how fast) 'error' $e^k=w^k-\overline{w}$ between the weight after iteration k, $w^k=p_k(M)w^0$ and the corresponding average load $\overline{w}=\frac{1}{n}(\sum_{i=1}^n w_i^0)(1,\ldots,1)^t$ tends to zero. These errors e^k have two fundamental properties, which are stated in the next lemma.

Lemma 2 ([DFM99]): *Let* $w^0=\sum_{i=1}^n z_i$ *as in Lemma 1. Then,* $e^0=\sum_{i=2}^n z_i$ *and* $e^k=p_k(M)e^0$ *for any* $k=0,1,2,\ldots$*.*

Using Lemma 2, we get

$$\|e^k\|_2\ \le\ \max_{i=2}^{n}|p_k(\mu_i)|\cdot\|e^0\|_2\ . \tag{2.3}$$

Here, $\|e^k\|_2$ and $\|e^0\|_2$ represent the vectors e^0 resp. e^k in l_2-norm. We take the first order-scheme (FOS) of Cybenko [Cyb89], where we have $p_k(t) = t^k$. These polynomials satisfy the simple short recurrence $p_k(t) = t \cdot p_{k-1}(t)$, $k = 1, 2, \ldots,$ so that we get $w^k = Mw^{k-1}$, $k = 1, 2, \ldots$. Now $|p_k(\mu_i)| = |\mu_i^k| \le \gamma^k$ for $i = 2, \ldots, n$ and with equation (2.3) results in $\|e^k\|_2 \le \gamma^k \cdot \|e^0\|_2$.

In [GMS96], the authors considered the second order scheme and calculated the number of steps needed to ϵ-balance the system. A system is ϵ-balanced after the k^{th} iteration step iff $\|e^k\|_2 \le \epsilon \cdot \|e^0\|_2$. Using their results, the following lemma can be stated.

Lemma 3: *Let G be a graph and let L be its Laplacian. Let $M = I - \alpha L$ be the diffusion matrix with $\alpha = \frac{2}{\lambda_2+\lambda_n}$ and set $\beta = \frac{2}{1+\sqrt{1-\mu_2^2}}$. Then, FOS and SOS take $O(\frac{1}{1-\gamma^2} \cdot \ln(1/\epsilon))$ and $O(\frac{1}{\sqrt{1-\gamma^2}} \cdot \ln(1/\epsilon))$ steps, respectively, to ϵ-balance the system.*

In this lemma α and β are chosen such that the convergence rate of FOS and SOS is maximized. We can see that the SOS converges faster than FOS by almost a quadratic factor. The Chebyshev method can be regarded to perform asymptotically identical to SOS [DFM99].

It holds that $\gamma = \max\{|1-\alpha\lambda_2|, |1-\alpha\lambda_n|\}$. The minimum of γ is achieved for $1-\alpha\lambda_2 = -1 + \alpha\lambda_n$. We get for α the optimal value $\alpha = \frac{2}{\lambda_2+\lambda_n}$. Then, we have

$$\gamma = 1 - \frac{2}{\lambda_2 + \lambda_n}\lambda_2 = \frac{\lambda_n - \lambda_2}{\lambda_2 + \lambda_n} = \frac{1-\rho}{1+\rho},$$

where $\rho = \frac{\lambda_2}{\lambda_n}$ is the condition number of the Laplace matrix L.

The l_2 flow minimization problem is described as follows:

$$\text{minimize } \|x\|_2 \text{ over all } x \text{ with } Ax = w^0 - \overline{w}.$$

As shown in [DFM99], the flow of a polynomial-based load balancing scheme $w^k = p_k(M)w^0$ is minimal in the l_2-norm if for all $k = 1, 2, \ldots$ the polynomials $p \in \overline{\Pi}_k$ satisfy the 3-term recurrence relation

$$p_k(t) = (\sigma_k t - \tau_k)p_{k-1}(t) + \rho_k p_{k-2}(t) \text{ , with } \sigma_k - \tau_k + \rho_k = 1. \tag{2.4}$$

Obviously, FOS, SOS, and the Chebyshev method fulfill the condition of equation (2.4) and therefore, they compute an l_2 minimal flow.

3. SPECTRAL BOUNDS ON THE K-PARTITIONING OF GRAPHS

In this chapter, we analyze the classical spectral lower bound described in (1.1) and derive new bounds on the k-section width of graphs, by using spectral techniques. First, we state a Courant-Fisher-like inequality for the sum of the k smallest eigenvalues of the Laplacian of a graph and determine a necessary condition to obtain equality in (1.1). Next, we introduce the class of level structured graphs and compute new lower bounds w.r.t. their k-section width. In Section 3.3, we define some weighted and unweighted graphs, and show that for these graphs the new bounds are tight up to a constant factor. We conclude by considering Cartesian powers of some dense regular graphs. In order to obtain improved bounds on the k-section width of these graphs, we use methods from discrete mathematics and combinatorics.

3.1 Basic Results

Let $G = (V, E)$ be a graph and let $n = |V|$. A powerful tool to determine new lower bounds for the k-partitioning size of G, is the Courant-Fischer-like inequality described in Theorem 1. This inequality generalizes the well-known bound described by the Rayleigh-coefficient,

$$\lambda_2 = \min_{x_i \perp \mathbf{1}} \frac{\sum_{\{q,r\}\in E}(x_{i,q} - x_{i,r})^2}{\sum_{q\in V} x_{i,q}^2},$$

where $x_i, \mathbf{1} \in \mathbb{R}^n$, $\mathbf{1} = (1, \ldots, 1)^t$ and $x_{i,q}$ denotes the q^{th} entry of the vector x_i. Theorem 1 can be immediately proved by using the so-called Representation theorem of [Bol93]. Here, we present a new proof using the result of [HW53].

Theorem 1: *Let $G = (V, E)$ be a graph and let n, k be two integers with $|V| = n$ and assume that $k|n$ holds. Let $\lambda_1, \lambda_2, \ldots, \lambda_k$ be the k smallest eigenvalues of the Laplacian L of G. Then, it holds that*

$$\sum_{i=1}^{k} \lambda_i \leq \min_{x_1,\ldots,x_k} \left\{ \sum_{i=1}^{k} \frac{\sum_{\{q,r\}\in E}(x_{i,q} - x_{i,r})^2}{\sum_{q\in V} x_{i,q}^2} \;\middle|\; x_i \perp x_j \text{ for } i,j \in \{1\ldots k\},\ i \neq j \right\}. \quad (3.1)$$

Proof: Let R be a matrix with the elements $R_{i,j} = x_{\lfloor j/k \rfloor +1,i}$. The first n/k columns of this matrix consist of the vector x_1, the next n/k columns consist of the vector x_2,..., and the last n/k columns consist of the vector x_k. In order to compute the trace of the matrix LRR^T, we first compute the elements of $L \cdot R$:

$$(LR)_{1,1} = \cdots = (LR)_{1,n/k} = x_{1,1} deg_1 - \sum_{(1,j)\in E} x_{1,j},$$

where deg_1 denotes the degree of the vertex $1 \in V$. Moreover

$$(LR)_{1,r} = x_{\lfloor r/k \rfloor+1,1} \cdot deg_1 - \sum_{(1,j)\in E} x_{\lfloor r/k \rfloor +1,j}$$

for all $1 \leq r \leq n$. It also holds, that

$$(LR)_{q,r} = x_{\lfloor r/k \rfloor+1,q} \cdot deg_q - \sum_{(q,j)\in E} x_{\lfloor r/k \rfloor +1,j}, \text{ with } q, r \in \{1, \ldots, n\}$$

Again, deg_q denotes the degree of the vertex $q \in V$. Multiplying (LR) with R^T we obtain

$$\begin{aligned}(LRR^T)_{q,q} &= \frac{n}{k}\Bigg(x_{1,q}^2 deg_q - \sum_{(q,j)\in E} x_{1,j}x_{1,q} + x_{2,q}^2 deg_q \\ &\quad - \sum_{(q,j)\in E} x_{2,j}x_{2,q} + \cdots + x_{k,q}^2 deg_q - \sum_{(q,j)\in E} x_{k,j}x_{k,q}\Bigg).\end{aligned}$$

Then,

$$Tr(LRR^T) = \frac{n}{k}\sum_{i=1}^{k}\sum_{(q,r)\in E}(x_{i,q} - x_{i,r})^2.$$

It is obvious that the spectrum of RR^T is the same as the spectrum of R^TR. Since $x_i \perp x_j$ for each pair i, j, $1 \leq i, j \leq k$ with $i \neq j$, the eigenvalues of RR^T are 0 with multiplicity $n-k$ and each $\frac{n}{k}\sum_{q\in V} x_{i,q}^2$ with multiplicity 1. Using the famous Hoffman-Wielandt theorem of [HW53], as described in [DH73], we get

$$\sum_{i=1}^{k}\sum_{(q,r)\in E}(x_{i,q} - x_{i,r})^2 \geq \lambda_1 \sum_{q\in V} x_{1,q}^2 + \cdots + \lambda_k \sum_{q\in V} x_{k,q}^2$$

where $\sum_{q\in V} x_{1,q}^2 \geq \sum_{q\in V} x_{2,q}^2 \geq \cdots \geq \sum_{q\in V} x_{k,q}^2$. We obtain the theorem by normalizing the vectors $x_1, x_2, \ldots, x_k$. □

Using the results of Theorem 1, we obtain the classical lower bound for the k-section width in the following way. Let $V = V_1 \uplus V_2 \uplus \ldots \uplus V_k$ be an optimal balanced k-partitioning of G. We consider the vectors $x_1, x_2, \ldots, x_k$ with the entries $x_{i,j} = 1$ whenever the vertex $j \in V_i$, and $x_{i,j} = 0$ otherwise. Using inequality (3.1) we obtain inequality (1.1).

In the following theorem we determine a necessary condition to have equality in (1.1).

Theorem 2: *Let $G = (V, E)$ be a graph, and let $V = V_1 \uplus V_2 \uplus \ldots \uplus V_k$ be an optimal balanced k-partitioning of G with cut size ∇, where $|V| = n$ and $k|n$. Let $\lambda_1, \lambda_2, \ldots, \lambda_k$ be the first k smallest eigenvalues of the Laplacian L of G and let $z_1, z_2, \ldots, z_k$ be corresponding eigenvectors. We represent by $z_{i,j}$ the j^{th} entry of z_i for any $1 \leq i \leq k$ and $1 \leq j \leq n$. Then, the statement*

a. $\nabla = \frac{n}{2k} \sum_{i=1}^{k} \lambda_i$
implies that

b. if $q, r \in V_i$ for an $i \in \{1, \ldots, k\}$, then $z_{j,q} = z_{j,r}$ for any $j \in \{1, \ldots, k\}$;
and

c. for any $i, j \in \{1, \ldots, k\}$, $i \neq j$ and any two vertices $q, r \in V_i$, the number of adjacent vertices to q in V_j equals the number of adjacent vertices to r in V_j.

Proof: We prove that statement (a) implies (b) and (b) implies (c).

(a) → (b): The Courant-Fischer theorem implies that by adding an additional edge to a graph G, no eigenvalue λ_i, $i \in \{1, \ldots, |V|\}$, of L can decrease (see equation (2.1)). For simplicity, we assume that all eigenvalues are simple. Let $\nabla = \frac{n}{2k} \sum_{i=1}^{k} \lambda_i$ and let $V = V_1 \uplus V_2 \uplus \ldots \uplus V_k$ be an optimal balanced k-partition of G. Assume that there exists an $i, j \in \{1, \ldots, k\}$ and two vertices $q, r \in V_i$ such that $z_{j,q} \neq z_{j,r}$ (Note, that z_j is an eigenvector of L with eigenvalue λ_j). We can assume w.l.o.g. that z_j is an eigenvector, which corresponds to the smallest eigenvalue among all eigenvalues having eigenvectors with this property. Then, it holds that $z_{1,q} = z_{1,r}$, $z_{2,q} = z_{2,r}$, $\ldots$, $z_{j-1,q} = z_{j-1,r}$. If there is no edge between q and r, then equation (2.1) implies that if we add edge $\{q, r\}$ to the graph G, then z_1, z_2, $\ldots$, z_{j-1} remain eigenvectors corresponding to the first $j-1$ eigenvalues of G with this new edge added to it, and λ_j increases without increasing the optimal cut of the k-partition. Thus, we have a contradiction to inequality (1.1).

If there is an edge between q and r, then we consider the Cartesian product $G \times K_n$ of G with the complete graph K_n. We denote with $w_1, \ldots, w_n$ the vertices of K_n. Then, the vertex (v, w_m), $m \in \{1, \ldots, n\}$ of $G \times K_n$ is adjacent to $(v', w_{m'})$, $m' \in \{1, \ldots, n\}$ iff $(v, v') \in E$ and $w_m = w_{m'}$ or $v = v'$. The eigenvalues of $G \times K_n$ have the form $\lambda_i + \nu_j$, where ν_j represent the eigenvalues of K_n [CDS95]. Since $\nu_1 = 0$ and $\nu_2 = \cdots = \nu_n = n$, the first k eigenvalues of $G \times K_n$ still remain $\lambda_1, \lambda_2, \ldots, \lambda_k$ and the classical lower bound will still be tight. Let y_j be an eigenvector of $G \times K_n$ corresponding to λ_j, $j \in \{1, \dots, k\}$. Furthermore, $y_{j,l,m}$ represents the entry of y_j corresponding to vertex (l, w_m) of $G \times K_n$. Then, it also holds that $y_{j,l,m} = z_{j,l}$ for any $m \in \{1, \dots, n\}$ and arbitrary j, l. Now, since we assumed that $z_{j,q} \neq z_{j,r}$ it follows that $y_{j,q,m} \neq y_{j,r,m'}$ for any $m, m' \in \{1, \dots, n\}$. By adding edge $\{(q, w_m), (r, w_{m'})\}$ to the graph $G \times K_n$, we increase λ_j and this leads to a contradiction to inequality (1.1).

In the case of multiple eigenvalues, if an eigenvector z_i corresponding to the eigenvalue $\lambda_i \neq \lambda_{k+1}$, $i \leq k$ does not satisfy property (b), then we get contradiction using the same arguments as in the case of simple eigenvalues. If an eigenvector z_i corresponding to an eigenvalue $\lambda_i = \lambda_{k+1}$, $i \leq k$ violates property (b), then we consider the following two cases. Let $s \in I\!N$ be the largest integer with the property $\lambda_s \neq \lambda_{k+1}$. If there exist $k - s$ eigenvectors $z_{s+1} \perp \cdots \perp z_k$, which satisfy property (b), then (a) implies (b), and the first statement of the theorem holds. Otherwise, we get a contradiction by applying the same arguments as in the case of simple eigenvalues.

(b) → (c): Assume that $z_{i,q} = z_{i,r}$ if $q, r \in V_j$ for any $i, j \in \{1, \dots, k\}$. Let z_i^j be the entry of z_i, which corresponds to any vertex $v \in V_j$ (so, $z_{i,q} = z_{i,r} = z_i^j$). Consider the system of linear equations

$$Z \cdot x_j \;=\; b^j, \tag{3.2}$$

where $Z \in I\!R^{k \times k}$, $Z = (z_i^s)$, $b^j \in I\!R^k$, $b^j = (\lambda_i z_i^j)$, and $x_j = (x_{j,s})$, $1 \leq j, s \leq k$ (x_j is the vector of indeterminants). Let $e_j = (e_{j,s})_{1 \leq s \leq k}$, where $e_{j,s}$ is the number of vertices in V_s adjacent to the vertex $v \in V_j$ if $s \neq j$, and $e_{j,j} = -\sum_{1 \leq s \leq k, s \neq j} e_{j,s}$ otherwise. Obviously e_j is a solution of (3.2). Since $z_1, \dots, z_k$ are pairwise perpendicular, Z has full rank and thus e_j is a unique solution. However, the system of linear equations (3.2) holds for all $u \in V_j$, and so (c) follows. □

Now, the question arises: Does *(c)* imply *(a)*? The answer is no. Consider for example the hypercube $Q(m)$ with $m \geq 4$, and let $k = 4$. The first three non-zero eigenvalues of $Q(m)$ are $\lambda_2 = \lambda_3 = \lambda_4 = 2$. The minimal cut of the 4-partition is 2^m and condition *(c)* of Theorem 2 is fulfilled. However, the lower bound described in inequality (1.1) results only in $2^m \frac{3}{4}$. Examples of graphs, for which the bound in (1.1) is tight are the complete graph or the complete bipartite graph. However, if condition *(c)* of theorem 2 holds, then there exist some eigenvalues $\lambda_{i_1}, \dots, \lambda_{i_k}$ (not necessarily the first k smallest) of G such that $\nabla = \frac{n}{2k} \sum_{j=1}^{k} \lambda_{i_j}$.

3.2 New Spectral Bounds on the k-Section Width of Level Structured Graphs

In Theorem 2, we have shown that if the lower bound described in inequality (1.1) is tight, then any vertex of G must be incident to a cut edge. Now, we consider the case where this condition is not satisfied, and show that for such graphs the lower bound can be improved significantly. This new result holds for arbitrary structures of the k-section and does not depend on the number and the positions of the cut-edges.

For the following, we consider the level structure of a k-section. Each level contains the vertices having the same distance to the cut.

Definition 3 (Level Structure): *Let $V = V_1 \uplus V_2 \uplus \ldots \uplus V_k$ be a k-section of a graph $G = (V, E)$ with $n = |V|$ and assume that $k|n$. We define the subsets V_i^j of V_i, $i \in \{1, \ldots, k\}$ as follows: Let V_i^1 be the set of all vertices in V_i, which are incident to a cut edge. Let V_i^j be the set of all vertices in V_i having distance $j-1$ to V_i^1. Furthermore, we denote with E_i^j, $j \geq 1$, $i \in \{1, \ldots, k\}$, the sets of edges, which connect vertices of V_i^j with vertices of V_i^{j+1}.*

Let $g : I\!N \to I\!N$ be a function. We denote with $LS(g, \nabla, k)$ the class of graphs having a k-section with a cut size ∇ and a level structure such that $|E_i^j| \leq \nabla g(j)$ for all $j \geq 1$ and $i \in \{1, \ldots, k\}$.

See Figure 3.1 for an example of a graph belonging to $LS(g, \nabla, 2)$. In a certain sense, the level structure can be viewed as k cones where the cone ends are connected by some edges. The function g represents their widths. For graphs with large cone width, information can flow more easily to the distant vertices and therefore, we can view this as a global expansion property. In the following, we show that for a fixed k-section width the spectral lower bound increases with the width of the cone.

In Lemma 4, we bound $\sum_{i=1}^k \lambda_i$ from above by some expression depending only on the growth function $g(j)$.

Lemma 4: *Let $G \in LS(g, \nabla, k)$ and let $l \in I\!N$ be an integer such that $n > k\nabla \sum_{j=1}^{l-1} g(j-1)$. Then,*

$$\sum_{i=1}^k \lambda_i \leq \min_{1=a_1 \leq a_2 \leq \cdots \leq a_l} \frac{2\nabla a_1^2 + k\nabla \sum_{j=1}^{l-1} g(j)(a_j - a_{j+1})^2}{\nabla \sum_{j=1}^{l-1} a_j^2 g(j-1) + a_l^2 \left(\frac{n}{k} - \nabla \sum_{j=1}^{l-1} g(j-1)\right)}.$$

Proof: We define some vectors $x_1, \ldots, x_k$ such that $x_i \perp x_j$ for any $1 \leq i, j \leq k$, $i \neq j$.
We choose the vectors x_i with entries $x_{i,v}$ for $v \in V$, defined by

$$x_{i,v} = \begin{cases} a_j, & \text{if } v \in V_i^j \text{ and } j < l \\ a_l, & \text{if } v \in V_i^j \text{ and } j \geq l \\ 0, & \text{otherwise} \end{cases} \tag{3.3}$$

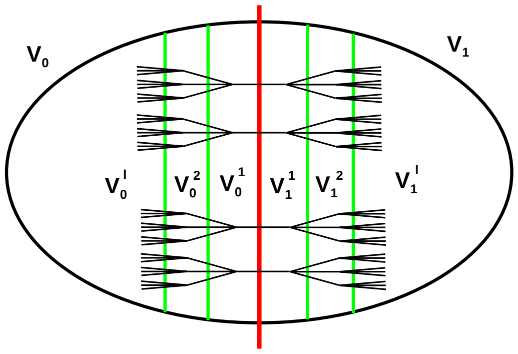


Fig. 3.1: Levels in a graph belonging to $LS(g, \nabla, 2)$. The edges connecting two consecutive levels increase with the function g.

where $1 = a_1 \leq a_2 \leq \cdots \leq a_l$. The upper bound of (3.1) depends only on the a_j's. We denote with $A(x_i) := \sum_{\{u,v\}\in E}(x_{i,u} - x_{i,v})^2$ the numerators in (3.1). Let $Q_{i,m}$ be the number of edges separating V_i from V_m for any $i, m \in \{1, \dots, k\}$. One has

$$\begin{aligned} A(x_i) &= \sum_{m=1}^{k} Q_{i,m} a_1^2 + \sum_{j=1}^{l-1} |E_i^j|(a_j - a_{j+1})^2 \qquad (3.4) \\ &\leq \sum_{m=1}^{k} Q_{i,m} a_1^2 + \nabla \sum_{j=1}^{l-1} g(j)(a_j - a_{j+1})^2. \end{aligned}$$

Note that $Q_{i,i} = 0$ for any $i \in \{1, \dots, k\}$.

To estimate the denominators $B(x_i) := \sum_{v\in V} x_{i,v}^2$ of inequality (3.1), assume that $|V_i^j| < |E_i^{j-1}|$ for some j, $1 \leq j < l$. Let $\tilde{x}_i$ be the vector obtained by moving a vertex from V_i^s, $s \geq l$ to V_i^j. One has $B(\tilde{x}_i) - B(x_i) = a_j^2 - a_l^2 < 0$. Therefore, $B(x_i)$ will not increase under this transformation. Since $|V_i^j| \leq |E_i^{j-1}|$, the minimum of $B(x_i)$ (for a fixed part V_i) is obtained if $|V_i^j| = |E_i^{j-1}| \leq \nabla g(j-1)$ for $1 \leq j < l$. By summarizing $\frac{A(x_i)}{B(x_i)}$ over all $i \in \{1, \dots, k\}$ we obtain

$$\sum_{i=1}^{k} \lambda_i \;\leq\; \sum_{i=1}^{k} \frac{A(x_i)}{B(x_i)} \leq \frac{2\nabla a_1^2 + k\nabla \sum_{j=1}^{l-1} g(j)(a_j - a_{j+1})^2}{\nabla \sum_{j=1}^{l-1} a_j^2 g(j-1) + a_l^2 \left(\frac{n}{k} - \nabla \sum_{j=1}^{l-1} g(j-1)\right)}$$

and the lemma follows. □

Lemma 4 shows that the level structure of the k-section gives an upper bound on $\sum_{i=1}^{k} \lambda_i$. In fact, the proof of the lemma shows that the worst case occurs if $|V_i^j| =$

$\nabla \cdot g(j-1)$ holds for any level j. We will use Lemma 4 in the following theorem to derive new relations between $\sum_{i=1}^{k} \lambda_i$ and the cut size ∇ of a k-section. These relations depend on the growth of the function g.

Theorem 3: *Let $G \in LS(g, \nabla, k)$. There exists a function $\gamma : I\!R^+ \to I\!R$ with $\gamma(x) \to 0$ for $x \to \infty$ such that*

- *If $A_g := 1 + \frac{2}{k}\sum_{j=2}^{\infty} \frac{1}{g(j-1)}$ and $\sum_{j=2}^{\infty} \frac{j-1}{g(j-1)} < \infty$, then*
$$\nabla \geq A_g \frac{\sum_{i=1}^{k} \lambda_i n}{2k}(1 - \gamma(\frac{n}{\nabla})).$$
- *Let LambertW(x) be the inverse function of $x \cdot e^x$. If $g(j) = (j+1)$, then*
$$\nabla \geq LambertW\left(\frac{4k}{\sum_{i=1}^{k} \lambda_i}\right) \cdot \frac{\sum_{i=1}^{k} \lambda_i n}{2k^2}(1 - \gamma(\frac{n}{\nabla})),$$
- *If $g(j) = (j+1)^\alpha$ and $0 \leq \alpha < 1$, then there exists a constant $\delta(\alpha)$, such that*
$$\nabla \geq \delta(\alpha)(\sum_{i=1}^{k} \lambda_i)^{\frac{\alpha+1}{2}} n(1 - \gamma(\frac{n}{\nabla})).$$

Proof: Let $l \in I\!N$ be defined by $k\nabla \sum_{j=1}^{l-1} g(j-1) < n \leq k\nabla \sum_{j=1}^{l} g(j-1)$. This, in particular, implies that $\frac{n}{\nabla} \to \infty$ as $l \to \infty$. We apply Lemma 4 with $a_j = 1 + \frac{2}{k}\sum_{m=2}^{j} \frac{1}{g(m-1)}$. Since $2a_1^2 + k\sum_{j=1}^{l-1} g(j)(a_j - a_{j+1})^2 = 2 + \frac{4}{k}\sum_{j=1}^{l-1} \frac{1}{g(j)} = 2a_l$, we obtain
$$\sum_{i=1}^{k} \lambda_i \leq \frac{2\nabla a_l}{\nabla \sum\limits_{j=1}^{l-1} a_j^2 g(j-1) + a_l^2 \left(\frac{n}{k} - \nabla \sum\limits_{j=1}^{l-1} g(j-1)\right)} \leq \frac{2a_l}{\sum\limits_{j=1}^{l-1} a_j^2 g(j-1)}. \tag{3.5}$$

In the following, we use some functions γ_1, γ_2, γ_3 and γ_4 with the property described in this theorem for the function γ.

- Assume $A_g := 1 + \frac{2}{k}\sum_{j=2}^{\infty} \frac{1}{g(j-1)} < \infty$: Set $r_j := A_g - a_j$, $1 \leq j \leq l$, with $r_j > 0$. The first inequality of (3.5) implies

$$\begin{aligned}\sum_{i=1}^{k} \lambda_i &\leq \frac{2\nabla(A_g - r_l)}{\nabla \sum\limits_{j=1}^{l-1} (A_g - r_j)^2 g(j-1) + (A_g - r_l)^2 \left(\frac{n}{k} - \nabla \sum\limits_{j=1}^{l-1} g(j-1)\right)} \\ &\leq \frac{2\nabla(A_g - r_l)}{\nabla A_g^2 \sum\limits_{j=1}^{l-1} (1 - \frac{2r_j}{A_g}) g(j-1) + A_g^2 (1 - \frac{2r_l}{A_g}) \left(\frac{n}{k} - \nabla \sum\limits_{j=1}^{l-1} g(j-1)\right)}\end{aligned}$$

We know that $\lim_{l\to\infty} \frac{2r_l}{A_g} = 0$. Now, we have to compute an upper bound for $\lim_{l\to\infty} \sum_{j=1}^{l-1} \frac{2r_j}{A_g} g(j-1)$. Using the Chebyshev inequalities, we obtain

$$\sum_{j=1}^{l-1} \frac{2r_j}{A_g} g(j-1) \le \frac{\sum_{j=1}^{l-1} 2r_j \cdot \sum_{j=1}^{l-1} g(j-1)}{A_g \cdot l}.$$

Then, $\sum_{j=1}^{l-1} \frac{2r_j}{A_g} = \frac{2}{A_g} \sum_{j=1}^{l-1} \frac{j-1}{g(j-1)}$ and therefore, a function γ_1 exists such that $\sum_{j=1}^{l-1} \frac{2r_j}{A_g} g(j-1) \le \gamma_1(l) \sum_{j=1}^{l-1} g(j-1)$. Then, we have

$$\sum_{i=1}^{k} \lambda_i \quad = \quad \frac{2\nabla(A_g - r_l)}{\nabla A_g^2(1-\gamma_1(l)) \sum\limits_{j=1}^{l-1} g(j-1) + A_g^2(1-\gamma_2(l)) \left(\frac{n}{k} - \nabla \sum\limits_{j=1}^{l-1} g(j-1) \right)}$$

It follows:

$$\sum_{i=1}^{k} \lambda_i \le \frac{2k\nabla}{A_g \cdot n(1-\gamma(l))}.$$

- Assume $g(j) = (j+1)$: Note that $\ln(n+1) = \int_1^{n+1} \frac{1}{x} dx \le \sum_{j=1}^{n} \frac{1}{j} \le 1 + \int_1^n \frac{1}{x} dx = 1 + \ln(n)$. Therefore, the second inequality in (3.5) with $g(j) = j+1$ provides

$$\begin{aligned}
\sum_{i=1}^{k} \lambda_i \quad &\le \quad \frac{2a_l}{\sum\limits_{j=1}^{l-1} j \cdot a_j^2} \le \frac{\frac{4}{k}\ln(l)(1+\gamma_1(l))}{1 + \sum\limits_{j=2}^{l-1} j(\frac{2}{k}\ln(j+1) + \frac{k-2}{k})^2} \\
&\le \quad \frac{\frac{4}{k}\ln(l)(1+\gamma_1(l))}{1 + \int\limits_1^{l-1} (\frac{4}{k^2} x \ln^2(x+1)(1+\gamma_2(\frac{n}{\nabla})))dx} \\
&\le \quad \frac{2k}{l^2 \ln(l)(1-\gamma_3(\frac{n}{\nabla}))} \; .
\end{aligned}$$

It follows from $n \le k\nabla \sum_{j=1}^{l} j = k\nabla \frac{l^2}{2}(1+\gamma_3(\frac{n}{\nabla}))$ that $l \ge \sqrt{\frac{2n}{k\nabla}(1+\gamma_3(\frac{n}{\nabla}))}$. This leads to $\sum_{i=1}^{k} \lambda_i \le \frac{k^2\nabla}{n\ln(\sqrt{2n/k\nabla})}(1+\gamma_3(\frac{n}{\nabla}))$. Solving this equation in ∇, we obtain the theorem.

- Assume $g(j) = (j+1)^\alpha$ with $0 \le \alpha < 1$: Since $\sum_{j=1}^{l} j^\beta = \frac{l^{\beta+1}}{\beta+1}(1+\gamma_1(l))$ for

$\beta > -1$, it holds that

$$\begin{aligned}\sum_{i=1}^{2}\lambda_i &\leq \frac{2a_l}{\sum\limits_{j=1}^{l-1} j^\alpha a_j^2} \leq \frac{\frac{4}{k}\cdot\frac{l^{1-\alpha}}{1-\alpha}}{\sum\limits_{j=1}^{l-1} j^\alpha \frac{4}{k^2}(\frac{j^{1-\alpha}}{1-\alpha})^2(1-\gamma_1(\frac{n}{\nabla}))} \\ &= \frac{k(1-\alpha)l^{1-\alpha}}{\sum\limits_{j=1}^{l-1} j^{2-\alpha}(1-\gamma_1(\frac{n}{\nabla}))} \leq \frac{k(1-\alpha)(3-\alpha)}{l^2(1-\gamma_2(\frac{n}{\nabla}))}.\end{aligned}$$

From $n \leq k\nabla\sum_{j=1}^{l} j^\alpha = k\nabla\frac{l^{\alpha+1}}{\alpha+1}(1+\gamma_3(\frac{n}{\nabla}))$ it follows that $l \geq (\frac{n(\alpha+1)}{k\nabla(1+\gamma_3(\frac{n}{\nabla}))})^{\frac{1}{\alpha+1}}$. This leads to

$$\sum_{i=1}^{k}\lambda_i \leq \frac{k(1-\alpha)(3-\alpha)}{(\frac{n(\alpha+1)}{k\nabla})^{\frac{2}{\alpha+1}}(1-\gamma_4(\frac{n}{\nabla}))} \quad \text{, i. e.}$$

$$\nabla \geq \frac{(1+\alpha)\cdot\left(\sum_{i=1}^{k}\lambda_i\right)^{\frac{\alpha+1}{2}} n}{k(k(1-\alpha)(3-\alpha))^{\frac{\alpha+1}{2}}}(1-\gamma(\frac{n}{\nabla})).$$

□

Let us analyze the 4-partitioning of a $\sqrt{n}\times\sqrt{n}$ torus. This graph belongs to $LS(g, 4\cdot\sqrt{n}, 4)$ with $g(x) = 1$ (see Figure 3.2 for the optimal balanced 4-section of an 8×8-torus). The sum of the first three non-zero eigenvalues is about $\frac{12\pi^2}{n}$ and the upper bound computed by Theorem 3 is $\frac{12\cdot(16)^2}{n}$. They differ only in a constant factor.

For an other important application of Theorem 3, we consider a graph G of maximum degree d. Then, $G \in LS(g,\nabla,k)$ with $g(j) = (d-1)^j$. Thus, Theorem 3 implies

$$\begin{aligned}\nabla &\geq A_g\frac{\sum_{i=1}^{k}\lambda_i n}{2k}(1-o(1)) \\ &= \left(1+\frac{2}{k}\left(\sum_{j=1}^{\infty}\frac{1}{(d-1)^j}\right)\right)\frac{\sum_{i=1}^{k}\lambda_i n}{2k}(1-o(1)) \\ &= \frac{k(d-2)+2}{k(d-2)}\cdot\frac{\sum_{i=1}^{k}\lambda_i n}{2k}(1-o(1)), \text{ as } l\to\infty.\end{aligned}$$

Using the results of [BEM$^+$00] w.r.t. the bisection width of Level Structured graphs, new spectral bounds on the bisection width of Ramanujan graphs could also be obtained. See [BEM$^+$00] for details.

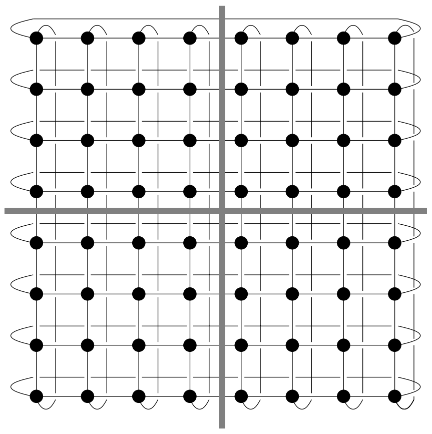


Fig. 3.2: The optimal 4-section of an 8×8-torus.

3.3 Lower Bounds on the Eigenvalues of Special Graph Classes

In this section we present several graphs, for which the bounds described in the previous section are asymptotically tight. First, we introduce a new class of edge-weighted graphs. They belong to the class $LS(g, \nabla, k)$ with $g(j) = (j+1)^\alpha$. Second, we analyze the k-rooted trees as a more realistic example.

3.3.1 Edge-Weighted Level Structured Graphs

As described in Chapter 2, the Laplacian of a graph can be generalized for edge-weighted graphs, where the off-diagonal entries contain the negative values of the corresponding edge-weights. The results of the previous sections and, especially, Theorem 3 can also be easily generalized to edge weighted graphs.

Definition 4: *We denote with $B^\alpha_{l,k}$, $\alpha \geq 0$ and $l \geq 1$, the edge-weighted graph obtained as follows. $B^\alpha_{l,k}$ consists of k isomorphic subgraphs. Each of these subgraphs contains l levels, with j^α vertices in level j, $1 \leq j \leq l$. Edges connect every pair of vertices in consecutive levels. Each edge connecting a vertex on level j with a vertex on level $j+1$ is weighted with $\frac{1}{j^\alpha}$. Moreover, the vertex on level 1 of each subgraph is connected with the vertices on level 1 of all other subgraphs by an edge of weight $\frac{2}{k}$.*

The graphs $B^\alpha_{l,k}$ belong to the class $LS(g, k-1, k)$ with $g(j) = (j+1)^\alpha$. An example with $\alpha = 2$, $l = 3$ and $k = 4$ is presented in Figure 3.3.

First, we consider the case $k = 2$ and in the following, we denote the graph $B^\alpha_{l,2}$ with B^α_l (see Figure 3.4 for an example). Throughout this section, $\lambda_2(B^\alpha_l)$ denotes the second

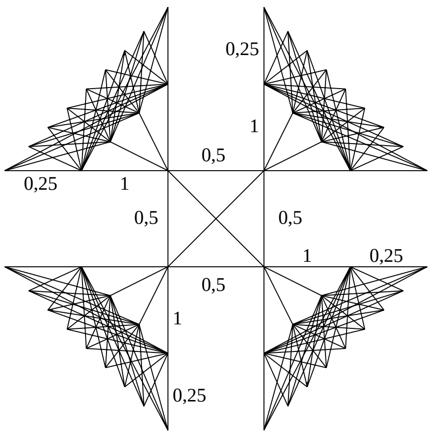


Fig. 3.3: The structure of the graph $B^2_{3,4}$. The values indicate the weight of the edges.

smallest eigenvalue of the Laplacian of B^α_l. We will show that the bounds calculated in theorem 3 differ only in a constant factor from the bisection width of these graphs. Note that similar results to Theorem 3 have been obtained in [BEM$^+$00] w.r.t. the bisection width of graphs. By setting $k = 2$ in Theorem 3, we observe that the bounds on the bisection width of [BEM$^+$00] are better than our bounds. However, the techniques used in [BEM$^+$00] can not be generalized for arbitrary k.

In order to compute $\lambda_2(B^\alpha_l)$, we need the following lemmas.

Lemma 5: *A matrix of the structure $\begin{pmatrix} M_1 & M_2 \\ M_2 & M_1 \end{pmatrix}$ has the same eigenvalues as the matrix $\begin{pmatrix} M_1+M_2 & 0 \\ 0 & M_1-M_2 \end{pmatrix}$, where M_1 and M_2 are square matrices of the same size.*

Proof: Let x be an eigenvector of $M_1 + M_2$ with eigenvalue λ. Then, $(x^t, x^t)^t$ is also an eigenvector of $\begin{pmatrix} M_1 & M_2 \\ M_2 & M_1 \end{pmatrix}$ with the same eigenvalue. If y is an eigenvector of $M_1 - M_2$ with eigenvalue λ', then $(y^t, -y^t)^t$ is an eigenvector $\begin{pmatrix} M_1 & M_2 \\ M_2 & M_1 \end{pmatrix}$ with the same eigenvalue λ'. $\square$

To prove Lemma 6, we need the so-called Separation theorem (see [Wil65]).

Theorem 4: *For a real symmetric matrix A_n, it holds that the eigenvalues $\lambda'_1, \ldots, \lambda'_{n-1}$ of any principal minor A_{n-1} of the matrix A_n separate the eigenvalues $\lambda_1, \ldots, \lambda_n$ of A_n.*

Now we are able to prove the following lemma.

Lemma 6: *If $\lambda_2(B^\alpha_l) < 2$, then the entries in the eigenvector of B^α_l, corresponding to $\lambda_2(B^\alpha_l)$, are equal for all vertices on the same level. Moreover $z^0_i = -z^1_i$, with z^s_i being the entry of the corresponding eigenvector on level i on side $s \in \{0, 1\}$.*

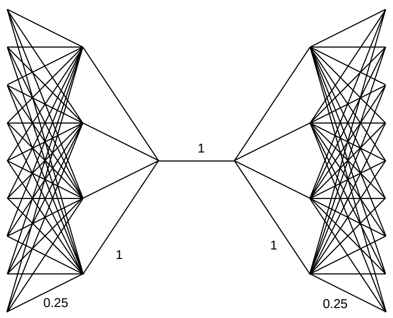


Fig. 3.4: The structure of the graph B_3^2. The values indicate the weight of the edges.

Proof: Consider two vertices $i1$ and $i2$ on level i on either side. Let z_{i1} and z_{i2} be their entries in the eigenvector corresponding to an eigenvalue λ of B_l^α, and assume $z_{i1} \neq z_{i2}$. Both vertices are adjacent to the same vertices on the levels $i-1$ and $i+1$. We denote with z_{iq} the entry in the eigenvector corresponding to a vertex q on level i, $1 \leq q \leq i^\alpha$. We obtain

$$-\sum_{q=1}^{(i-1)^\alpha} z_{(i-1)q} + deg_i z_{i\epsilon} - \sum_{q=1}^{(i+1)^\alpha} \frac{g(i)}{g(i-1)} z_{(i+1)q} = \lambda z_{i\epsilon} \text{ , i.e.}$$

$$(deg_i - \lambda) z_{i\epsilon} = \sum_{q=1}^{(i-1)^\alpha} z_{(i-1)q} + \sum_{q=1}^{(i+1)^\alpha} \frac{g(i)}{g(i-1)} z_{(i+1)q}$$

where $\epsilon \in \{1,2\}$. Then, $z_{i\epsilon} = 0$ or $\lambda = deg_i = 1 + (\frac{i+1}{i})^\alpha > \lambda_2$, where deg_i is the degree of a vertex on level i. If $z_{i1} \neq z_{i2}$, then the corresponding eigenvalue is not the second smallest of our Laplacian.

We arrange the vertices of the graph in the Laplacian in the following way. The first $n/2$ rows are filled with the vertices of the left side of the bisection, where we begin with the vertex, which is incident to the bisection edge. The next $n/2$ rows represent the vertices on the right hand of the bisection, and we use the same order for them as for the first $n/2$ vertices. Then the Laplacian of B_l^α has the structure $\begin{pmatrix} M_1 & M_2 \\ M_2 & M_1 \end{pmatrix}$, and its eigenvalues are the union of the eigenvalues of the matrices $M_1 + M_2$ and $M_1 - M_2$ (see Lemma 5). The first entry of M_2 consists of a -1 and all other entries are 0.

We denote with $(M_1 + M_2)(1,1)$ the matrix obtained from $M_1 + M_2$ by deleting its first row and first column; and denote with $(M_1 - M_2)(1,1)$ the matrix obtained from $M_1 - M_2$ by deleting its first row and first column. Using the Separation theorem, it is easy to see that the eigenvalues of $(M_1 + M_2)(1,1)$ separate the eigenvalues of $M_1 + M_2$ and the eigenvalues of $(M_1 - M_2)(1,1)$ separate the eigenvalues of

$M_1 - M_2$. However, $(M_1 + M_2)(1,1) = (M_1 - M_2)(1,1)$ and 0 is an eigenvalue of $(M_1 + M_2)$. Hence, the smallest eigenvalue of $M_1 - M_2$ is smaller than the second smallest eigenvalue of $M_1 + M_2$. Therefore, the second smallest eigenvalue of the Laplacian of B_l^α is the smallest eigenvalue of $M_1 - M_2$ and, as shown in Lemma 5, $z_i^1 = -z_i^0$ holds for all $1 \leq i \leq l$. □

Using Lemma 6, $\lambda_2(B_l^\alpha)$ can be obtained from the eigenvalues of a certain weighted path.

Now we are going to calculate lower bounds for $\lambda_2(B_l^\alpha)$. It follows from Theorem 3 that there exists an l_0 such that, for any $l \geq l_0$, it holds that $\lambda_2(B_l^\alpha) < 2$. In the following we consider $l \geq l_0$. We denote with z_i the entry of the eigenvector on the left side for level i (z corresponds to $\lambda_2(B_l^\alpha)$). We set $z_i = b_i + r_i$, where $r_i = -\lambda_2(B_l^\alpha)p_i$ and $b_i = 1 + 2\sum_{j=1}^{i-1} \frac{1}{g(j)}$. Here, r_i represents an error and, in the following, we show that the r_i's are small compared with b_i.

From the equality $(g(i-1) + g(i))z_i - g(i-1)z_{i-1} - g(i)z_{i+1} = \lambda_2(B_l^\alpha)g(i-1)z_i$, $2 \leq i \leq l-1$ and from $(g(1)+1)z_1 + z_1 - g(1)z_2 = \lambda_2(B_l^\alpha)z_1$, it follows that $p_{i+1} = p_i + \alpha(i)((1-\lambda_2)p_i - p_{i-1} + b_i)$, where $p_1 = 0$, $p_2 = \frac{1}{g(1)}$ and $\alpha(i) = \frac{g(i-1)}{g(i)}$.

First, we consider the case $\alpha > 1$ and let $A'_g = 1 + 2\sum_{j=1}^{\infty} \frac{1}{g(j)} < \infty$. Let q_i be defined by the equation $q_{i+1} = q_i + \alpha(i)(q_i - q_{i-1} + A'_g)$, where $q_1 = 0$ and $q_2 = \frac{1}{g(1)}$. From the definition of q_i, we obtain $q_i = \sum_{j=1}^{i-1} \frac{1}{g(j)} + \sum_{(p,q),1\leq p<q\leq i-1} \frac{g(p)}{g(q)} A'_g$ and $q_{i+1} - q_i = \frac{1}{g(i)} + \sum_{p=1}^{i-1} \frac{g(p)}{g(i)} A'_g$.

Lemma 7: *For the graph B_l^α with $\alpha > 1$, it holds that $q_{i+1} - q_i \geq p_{i+1} - p_i \geq 0$ for any $1 \leq i \leq l-1$. Moreover, $\lambda_2(B_l^\alpha)q_l \to \infty$*

Proof: We first show the second statement of our lemma.

$$\begin{aligned} q_i &= \sum_{j=1}^{i-1} \frac{1}{g(j)} + \sum_{1\leq p<q\leq i-1} \frac{g(p)}{g(q)} A'_g \leq A'_g + \sum_{q=1}^{i-1}\sum_{p=1}^{q-1} \frac{g(p)}{g(q)} A'_g \\ &\leq A'_g + \sum_{p=1}^{i-1}\left(\sum_{q=p+1}^{i-1} \frac{1}{g(q)}\right)g(p) \end{aligned}$$

Theorem 3 implies that $\lambda_2(B_l^\alpha) \leq O(l^{-2})$ and therefore, the second statement of the lemma holds. For the first statement let $\delta_i = q_i - p_i$. We use induction on i. For $i = 1$ the lemma holds. We assume that the lemma holds for any $j \leq i$. Then,

$$\begin{aligned} \delta_{i+1} &= q_{i+1} - p_{i+1} = q_i + \alpha(i)(q_i - q_{i-1} + A'_g) - p_i \\ &\quad -\alpha(i)((1-\lambda_2(B_l^\alpha))p_i - p_{i-1} + b_i) \\ &= \delta_i + \alpha(i)(\delta_i - \delta_{i-1}) + \alpha(i)(A'_g - b_i) + \alpha(i)(p_{i-1} + \lambda_2(B_l^\alpha)p_i) \geq \delta_i \end{aligned}$$

Now, we have to show that $p_{i+1} - p_i \geq 0$. Our assumption was that the lemma holds for any j, $0 \leq j \leq i$. This implies that $0 \leq p_i \leq q_i$. Then $p_{i+1} - p_i \geq \alpha(i)(b_i - \lambda_2(B_l^\alpha)q_i)(B_l^\alpha)\frac{i^2}{2\alpha+2}A'_g) \geq 0$, where $\lambda_2(B_l^\alpha) \leq O(l^{-2})$ because of theorem 3 and therefore the last inequality is true. □

In the next theorem, we state a lower bound for $\lambda_2(B_l^\alpha)$, where we assume that $A'_g = 1 + 2\sum_{j=1}^{\infty} \frac{1}{g(j)} < \infty$.

Theorem 5: *If $\alpha > 1$, then it holds*

$$\lambda_2(B_l^\alpha) \geq \frac{4}{A'_g} \cdot \frac{1}{n(1+o(1))}.$$

Proof: Let us consider $\lambda_2(B_l^\alpha) = \frac{f_1(z)}{f_2(z)}$, where eigenvector z corresponds to $\lambda_2(B_l^\alpha)$. Now, using Lemma 6, it holds that $f_1(z) = 4z_1^2 + 2\sum_{i=1}^{l-1}(z_{i+1} - z_i)^2 g(i)$ and $f_2(z) = 2\sum_{i=1}^{l} z_i^2 g(i-1)$. We get

$$\begin{aligned} f_1(z) &= 4 + 2\sum_{i=1}^{l-1}(2\frac{1}{g(i)} - \lambda_2(B_l^\alpha)(p_{i+1} - p_i))^2 g(i) \\ &\geq 4 + 2\sum_{i=1}^{l-1}\frac{4}{g(i)} - 8\lambda_2(B_l^\alpha)\sum_{i=1}^{l-1}(q_{i+1} - q_i) \geq 4b_l - o(l) \end{aligned}$$

and

$$f_2(z) \leq 2\sum_{i=1}^{l-1}(b_i^2 + \lambda_2^2(B_l^\alpha)q_i^2)g(i) \leq 2\sum_{i=1}^{l-1} b_i^2 g(i) + o(l^2) \leq 2\sum_{i=1}^{l-1} b_l^2 g(i) + o(l^2)$$

Calculating the bounds for $f_1(z)$ and $f_2(z)$, we obtain the result of the theorem. □

This theorem can be easily generalized for any function g, such that $A'_g < \infty$.

We are ready to discuss the case $0 \leq \alpha < 1$. We define q_i by the equation $q_{i+1} = q_i + \alpha_i(q_i - q_{i-1} + b_i)$, where $q_1 = 0$ and $q_2 = \frac{1}{g(1)}$. We obtain $q_i = \sum_{j=1}^{i-1}\frac{1}{g(j)} + \sum_{p=1}^{i-2}\sum_{q=p+1}^{i-1}\frac{g(p)}{g(q)}b_{p+1}$.

Lemma 8: *For $0 \leq \alpha < 1$, it holds that $q_i \leq \frac{1}{(1-\alpha)(3-\alpha)} i^{3-\alpha}(1 + O(i^{\alpha-1}))$. Moreover, $q_{i+1} - q_i \geq p_{i+1} - p_i \geq 0$ for a sufficiently large l, $1 \leq i \leq l-1$.*

Proof: First, we set $\sigma_i = \sum_{j=1}^{i-1} \frac{1}{(j+1)^\alpha} \leq \frac{1}{1-\alpha}(i+1)^{1-\alpha}$. We get

$$q_i = O(i^{1-\alpha}) + \sigma_i \sum_{p=1}^{i-2} g(p)b_{p+1} - \sum_{p=1}^{i-2} \sigma_{p+1} g(p) b_{p+1} = \frac{1}{(1-\alpha)(3-\alpha)} i^{3-\alpha}.$$

For the second statement of the lemma we use induction on i. If $i = 1$, then the lemma holds. We assume that the lemma holds for any $j \leq i$. Then, $q_{i+1} - q_i \geq p_{i+1} - p_i$ is equivalent to $q_{i+1} - p_{i+1} \geq q_i - p_i$. Now,

$$\begin{aligned} q_{i+1} - p_{i+1} &= q_i + \alpha(i)(q_i - q_{i-1} + b_i) - p_i - \alpha(i)((1 - \lambda_2(B_l^\alpha))p_i - p_{i-1} + b_i) \\ &= q_i - p_i + \alpha(i)((q_i - p_i) - (q_{i-1} - p_{i-1}) + \lambda_2(B_l^\alpha)p_i) \geq q_i - p_i \end{aligned}$$

We assume that the lemma holds for any j, $1 \leq j \leq i$, and therefore, $0 \leq p_i \leq q_i$. Then, we obtain

$$p_{i+1} - p_i = \alpha(i)(p_i - p_{i-1} + b_i - \lambda_2(B_l^\alpha)p_i) = \alpha(i)(p_i - p_{i-1}) + \alpha(i)(b_i - \lambda_2(B_l^\alpha)p_i)$$

We have to show that $b_i - \lambda_2(B_l^\alpha)p_i \geq 0$. Using Lemma 8, we have

$$b_i - \lambda_2(B_l^\alpha)p_i \geq (1 - \frac{1-\alpha}{2-\alpha}\frac{i^2(1 + O(i^{\alpha-1}))}{n^2(1+o(1))})b_i \geq 0.$$

□

Now we are ready to calculate a lower bound for $\lambda_2(B_l^\alpha)$.

Theorem 6: *If $0 \leq \alpha < 1$, then there exists a constant $c(\alpha)$ such that $\lambda_2(B_l^\alpha) \geq c(\alpha)\frac{1}{l^2(1+o(1))}$, where $c(\alpha) = \frac{2(1-\alpha^2)(3-\alpha)(7-\alpha)}{4(7-\alpha)+(1-\alpha)^2(3-\alpha)}$.*

Proof: It holds that $\lambda_2(B_l^\alpha) = \frac{f_1(z)}{f_2(z)}$, where $f_1(z) \geq 4b_l - 4\lambda_2(B_l^\alpha)\sum_{i=1}^{l-1}(q_{i+1} - q_i)$ and $f_2(z) \leq 2\sum_{i=1}^{l-1}(b_i^2 + \lambda_2^2(B_l^\alpha)q_i^2)g(i)$. In the following we omit the low order terms and obtain

$$\begin{aligned} q_i - q_{i-1} &= \frac{1}{g(i-1)} + \sum_{p=1}^{i-2} \frac{g(p)}{g(i-1)} b_{p+1} \\ &= O(i^{-\alpha}) + \frac{1}{i^\alpha}\sum_{p=1}^{i-2}(p+1)^\alpha \frac{2}{1-\alpha}(p+1)^{1-\alpha} \\ &= O(i^{-\alpha}) + \frac{1}{i^\alpha}\frac{2}{1-\alpha}\frac{i^2}{2} = \frac{1}{1-\alpha} i^{2-\alpha} + O(i^{-\alpha}) \end{aligned}$$

and

$$\lambda_2(B_l^\alpha)\sum_{i=1}^{l-1}(q_{i+1}-q_i) \le \frac{(1-\alpha)(3-\alpha)}{l^2}\frac{1}{(1-\alpha)(3-\alpha)}l^{(3-\alpha)}.$$

Then, we get

$$f_1(z) \ge 4\left(\frac{2}{1-\alpha}l^{1-\alpha}-l^{1-\alpha}\right) = 4\frac{1+\alpha}{1-\alpha}l^{1-\alpha}.$$

On the other side, using Lemma 8, it holds that

$$\begin{aligned} f_2(z) &\le 2\sum_{i=1}^{l} b_i^2 g(i) + 2\lambda_2^2(B_l^\alpha)\sum_{i=1}^{l} q_i^2 g(i) \\ &= 2\left(\frac{4}{(1-\alpha)^2(3-\alpha)}l^{3-\alpha}+\frac{1}{l^4(7-\alpha)}l^{7-\alpha}\right). \end{aligned}$$

□

In the following, we consider the case $\alpha = 1$ and define q_i by the equation $q_{i+1} = q_i + \alpha_i(q_i - q_{i-1} + b_i)$, where $q_1 = 0$ and $q_2 = \frac{1}{g(1)}$. We obtain $q_i = \sum_{j=1}^{i-1}\frac{1}{g(j)} + \sum_{p=1}^{i-2}\sum_{q=p+1}^{i-1}\frac{g(p)}{g(q)}b_{p+1}$.

Lemma 9: *Let $g(i) = (i+1)$ and $\alpha = 1$. Then, it holds that $q_{i+1} - q_i \ge p_{i+1} - p_i \ge 0$ for any i, $1 \le i \le l-1$. Moreover, $0 \le q_i \le (1 + \frac{1}{4}i^2)b_i$ for any i, $1 \le i \le l-1$.*

Proof: We first show the second statement of our lemma.

$$q_i = \sum_{j=1}^{i-1}\frac{1}{g(j)} + \sum_{1\le p<q\le i-1}\frac{g(p)}{g(q)}b_i \le b_i + \sum_{q=1}^{i-1}\sum_{p=1}^{q-1}\frac{g(p)}{g(q)}b_i \le b_i + \sum_{q=1}^{i-1}\frac{q}{2}b_i \le b_i + \frac{i^2}{4}b_i$$

For the first statement let $\delta_i = q_i - p_i$. We use induction on i. For $i = 1$ the lemma holds. We assume that the lemma holds for any $j \le i$. Then,

$$\begin{aligned} \delta_{i+1} &= q_{i+1} - p_{i+1} = q_i + \alpha(i)(q_i - q_{i-1} + A'_g) - p_i \\ &\quad -\alpha(i)((1-\lambda_2(B_l^\alpha))p_i - p_{i-1} + b_i) \\ &= \delta_i + \alpha(i)(\delta_i - \delta_{i-1}) + \alpha(i)(A'_g - a_i) + \alpha(i)(p_{i-1} + \lambda_2(B_l^\alpha)p_i) \ge \delta_i. \end{aligned}$$

Now, we have to show that $p_{i+1} - p_i \ge 0$. Our assumption was that the lemma holds for any j, $0 \le j \le i$. This implies $0 \le p_i \le q_i$. Then, $p_{i+1} - p_i \ge \alpha(i)(b_i - \lambda_2(B_l^\alpha)q_i) \ge \alpha(i)(b_i - \lambda_2(B_l^\alpha)b_i - \lambda_2(B_l^\alpha)\frac{i^2}{4}b_i) \ge 0$, where $\lambda_2(B_l^\alpha) = O(\frac{1}{l^2\ln(l)})$ because of Theorem 3 and therefore the last inequality holds. □

Next, we state a lower bound for $\lambda_2(B_l^1)$ in the next theorem.

Theorem 7: *It holds that*

$$\lambda_2(B_l^1) \geq \frac{2}{l^2 \ln(l)(1+o(1))}.$$

Proof: Let us consider $\lambda_2(B_l^1) = \frac{f_1(z)}{f_2(z)}$, where eigenvector z corresponds to $\lambda_2(B_l^1)$. Using Lemma 6, we obtain that $f_1(z) = 4z_1^2 + 2\sum_{i=1}^{l-1}(z_{i+1} - z_i)^2 g(i)$ and $f_2(z) = 2\sum_{i=1}^{l} z_i^2 g(i-1)$. Then,

$$\begin{aligned} f_1(z) &= 4 + 2\sum_{i=1}^{l-1}(2\frac{1}{g(i)} - \lambda_2(B_l^1)(p_{i+1} - p_i))^2 g(i)) \\ &\geq 4 + 2\sum_{i=1}^{l-1}\frac{4}{g(i)} - 8\lambda_2(B_l^1)\sum_{i=1}^{l-1}(q_{i+1} - q_i) \geq 4\ln(l)(1-o(1)) \end{aligned}$$

and

$$f_2(z) \leq 2\sum_{i=1}^{l-1}(b_i^2 + \lambda_2^2(B_l^1)q_i^2)g(i) \leq 2\sum_{i=1}^{l-1} b_i^2 g(i) + o(1) \leq 2l^2\ln^2(l)(1+o(1))$$

Calculating the bounds for $f_1(z)$ and $f_2(z)$, we obtain the result of the theorem. □

Let us compare the results of the theorems 5, 6 and 7 with the results of Theorem 3. For the case $\alpha > 1$, the upper and lower bounds on λ_2 of theorems 3 and 5 differ in a factor of $\frac{1+2\sum_{j=1}^{\infty}\frac{1}{g(j)}}{1+\sum_{j=1}^{\infty}\frac{1}{g(j)}} < 2$. For the case $\alpha = 1$, this factor equals 2. In the case $0 < \alpha < 1$, the bounds of theorems 3 and 6 are tight up to a constant factor. To see this, it is known that $n = 2\sum_{i=1}^{l} g(i) = \frac{2}{\alpha+1} l^{\alpha+1}(1+o(1))$ and thus, there exits a $d(\alpha)$ such that $\lambda_2(B_l^\alpha) \geq d(\alpha)\frac{1}{n^{\frac{2}{\alpha+1}}(1+o(1))}$. A special case is $\alpha = 0$. The graph B_l^0 is a path of length $2l$ and it holds that $c(0) = \frac{42}{31}$ from Theorem 6. From Theorem 3 we get $\lambda_2 \leq 6\frac{1}{l^2(1+o(1))}$. However, for the path we know that $\lambda_2 \approx \frac{\pi^2}{4l^2}$. Thus, there is a gap of a constant factor between the lower and the upper bound for λ_2.

In the following, we use the results of theorems 5-7 to compute the k smallest eigenvalues of $B_{l,k}^\alpha$ for arbitrary k's.

Theorem 8: *Denoting with $\lambda_i(B_{l,k}^\alpha)$, $1 \leq i \leq k$, the smallest k eigenvalues of the Laplacian of $B_{l,k}^\alpha$, there exist some constants $c_1(\alpha)$ and $c_2(\alpha)$ such that*

$$\lambda_i(B_{l,k}^\alpha) = \begin{cases} \frac{2k}{n(1+2\sum_{j=2}^{\infty}\frac{1}{g(j-1)})}(1+o(1)) & \text{if } \alpha > 1 \\ c_1(\alpha)\frac{1}{n\ln(n)}(1+o(1)) & \text{if } \alpha = 1 \\ c_2(\alpha)\left(\frac{1}{n}\right)^{\frac{2}{\alpha+1}}(1+o(1)) & \text{if } 0 \leq \alpha < 1. \end{cases}$$

Proof: To compute the eigenvalues $\lambda_i(B^{\alpha}_{l,k})$, we use the following observation. The eigenvalues of a square matrix of the form

$$\begin{pmatrix} M_1 & M_3 & M_3 & \dots & M_3 \\ M_3 & M_1 & M_3 & \dots & M_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ M_3 & M_3 & M_3 & \dots & M_3 \\ M_3 & M_3 & M_3 & \dots & M_1 \end{pmatrix} \tag{3.6}$$

are the eigenvalues of the matrix

$$M_1 + \sum_{j=1}^{k-1} (\sqrt[k]{1})^{(i-1)\cdot j} \cdot M_3 = \begin{cases} M_1 + (k-1)\cdot M_3 & \text{if } i = 1 \\ M_1 - M_3 & \text{otherwise} \end{cases}$$

where $\sqrt[k]{1} = \cos(\frac{2\pi}{k}) + I \cdot \sin(\frac{2\pi}{k})$ (see [Dav79]). Here, M_1 and M_3 are square matrices. Then it follows that the eigenvalues of $M_1 - M_3$ appear $k-1$ times as eigenvalues of the original matrix.

Now, the Laplacian of $B^{\alpha}_{l,k}$ has the form of the matrix in (3.6), where M_1 represents one subgraph of $B^{\alpha}_{l,k}$ and M_3 represents a single edge, which separates one subgraph from another in $B^{\alpha}_{l,k}$. Hence, $M_{3(1,1)} = -\frac{2}{k}$, and all other entries of M_3 are 0; furthermore, M_1 contains the Laplacian of a subgraph of $B^{\alpha}_{l,k}$, where the value $\frac{2(k-1)}{k}$ is added to $M_{1(1,1)}$. Now the first k smallest eigenvalues of $B^{\alpha}_{l,k}$ are as follows: 0 appears once, and the smallest eigenvalue of $M_1 - M_3$ appears $k-1$ times. However, the smallest eigenvalue of the matrix $M_1 - M_3$ is the second smallest eigenvalue of the Laplacian of the graph B^{α}_{l} already computed and the theorem follows. □

The example of the graph $B^{\alpha}_{l,k}$ shows that the bounds calculated in Theorem 3 are tight up to a constant factor. Again, we concentrate on the case $\alpha > 1$. The upper bound on $\sum_{i=1}^{k} \lambda_i(B^{\alpha}_{l,k})$ obtained from Theorem 3 and the value computed in theorem 8 differ in a factor of

$$\frac{1 + 2\sum_{j=2}^{\infty} \frac{1}{g(j-1)}}{1 + \frac{2}{k}\sum_{j=2}^{\infty} \frac{1}{g(j-1)}}$$

from each other.

In the other cases, the quotient between the upper and the lower bound results in a more complicated formula, however, they differ only in a constant factor from each other. In this subsection, we presented edge-weighted graphs, in order to show that the bounds, computed in the main theorem of this chapter, can be tight for large graph classes. In the next subsection, we consider graphs without edge-weights.

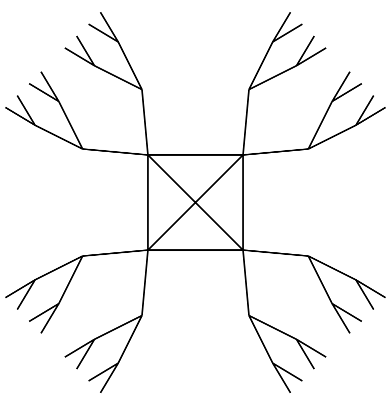


Fig. 3.5: The structure of the graph $T_{3,4,4}$.

3.3.2 The k-Rooted Tree

Another example of level structured graphs are k-rooted trees defined as follows (see Figure 3.5).

Definition 5: *The k-rooted tree $T_{d,l,k}$ consists of k complete $(d-1)$-ary subtrees, and every vertex of these subtrees at distance $1, 2, \ldots, l-2$ from the root has degree d. The vertices, which are at distance $l-1$ from the root have degree 1. The roots of the trees are pairwise connected with each other, and therefore the degree of these roots equals $(k-1)+(d-1)$.*

We compute the k smallest eigenvalues of a k-rooted tree and show that the spectral lower bound provided by Theorem 3 differs from the k-section width only in a small constant factor. We denote with $\lambda_1, \lambda_2, \ldots, \lambda_k$ the k smallest eigenvalues of the Laplacian of the k-rooted tree $T_{d,l,k}$. Furthermore, let $B(T_{d,l,k})$ be the adjacency matrix of a weighted k-rooted tree, in which each leaf has a loop of weight $d-1$ and each root has a loop of weight $2-k$. Denoting with ν_i the eigenvalues of $B(T_{d,l,k})$, it follows that $\lambda_i = d - \nu_{n-i+1}$.

Therefore, it is sufficient to concentrate on the computation of the eigenvalues of $B(T_{d,l,k})$. Using the techniques of theorem 8, we get $\nu_n = d$. Furthermore, we obtain that $\nu_{n-1} = \ldots = \nu_{n-k+1}$ is the largest eigenvalue of the adjacency matrix of a weighted $(d-1)$-ary tree with height $l-1$ and having a loop of weight $1-k$ at the root and a loop of weight $d-1$ at the leafs. As shown in [Til99], the largest eigenvalue of such a matrix equals the largest eigenvalue of the adjacency matrix of a weighted path of length l having a loop of weight $1-k$ at one end and a loop of weight $d-1$ at the other end. The edges between two consecutive vertices have weight $\sqrt{d-1}$.

We denote with $Q(X)$ the characteristic polynomial of the adjacency matrix corresponding to such a weighted path. Then,

$$Q(X) = (X + (k-1))P_{l-1}(X) - (d-1)P_{l-2}(X),$$

where $P_i(X)$ is the characteristic polynomial of the adjacency matrix corresponding to the weighted path of length i, described above, without the loop of weight $1-k$ at the one end. As described in [Til99], it can be easily checked that, if we set $X = 2\sqrt{d-1}\cosh(\phi)$, then

$$P_l(X) = XP_{l-1}(X) - (d-1)P_{l-2}(X) = (d-1)^{l/2}\frac{\sinh((l+1)\phi) - \sqrt{d-1}\sinh(l\phi)}{\sinh(\phi)}. \quad (3.7)$$

Now we are ready to state the following theorem.

Theorem 9: *For large l, the first $k-1$ nonzero eigenvalues of the Laplacian of the graph $T_{d,l,k}$ have the form*

$$\lambda_i = \left(\frac{k(d-2)^2}{d+k-2} - o(1)\right)\frac{1}{(d-1)^l}.$$

Proof: Using equation (3.7), we obtain

$$\begin{aligned}
Q(X) &= XP_{l-1}(X) - (d-1)P_{l-2}(X) + (k-1)P_{l-1}(X) \\
&= P_l(X) + (k-1)P_{l-1}(X) \\
&= \frac{(d-1)^{l/2}}{\sinh(\phi)}\Big(\sqrt{d-1}\cdot\sinh((l+1)\phi) - (d-1)\sinh(l\phi) \\
&\quad + (k-1)\sinh(l\phi) - (k-1)\sqrt{d-1}\cdot\sinh((l-1)\phi)\Big) \\
&= \frac{(d-1)^{l/2}}{\sinh(\phi)}\Big(\sqrt{d-1}(\sinh(l\phi)\cosh(\phi) + \sinh(\phi)\cosh(l\phi)) \\
&\quad - (d-k)\sinh(l\phi) - (k-1)\sqrt{d-1}(\sinh(l\phi)\cosh(\phi) - \sinh(\phi)\cosh(l\phi))\Big) \\
&= \frac{(d-1)^{l/2}\cosh(l\phi)}{\sinh(\phi)}\Big(\sqrt{d-1}(\tanh(l\phi)\cosh(\phi) + \sinh(\phi)) \\
&\quad - (d-k)\tanh(l\phi) - (k-1)\sqrt{d-1}(\tanh(l\phi)\cosh(\phi) - \sinh(\phi))\Big) \\
&= \frac{(d-1)^{l/2}\cosh(l\phi)}{\sinh(\phi)}\Big(\sqrt{d-1}(2-k)\tanh(l\phi)\cosh(\phi) \\
&\quad - (d-k)\tanh(l\phi) + \sqrt{d-1}\cdot k\cdot\sinh(\phi)\Big)
\end{aligned}$$

Let $Q'(X)$ be defined by

$$Q(X) = \frac{(d-1)^{l/2}\cosh(l\phi)}{\sinh(\phi)} Q'(X).$$

Obviously $Q(X) = 0$ iff $Q'(X) = 0$. Theorem 3 implies that $2\sqrt{d-1}\cosh(\phi) = d - O(\frac{1}{(d-1)^l})$. Since $\lim_{x\to\infty} \tanh x = 1 - (2+o(1))e^{-2x}$, we get

$$\tanh(l\phi) = 1 - (2+o(1))(d-1+o(1))^{-l} = 1 - 2(d-1)^{-l} - o(1)(d-1)^l. \quad (3.8)$$

Using equation (3.8) and the fact that $\sinh(\phi) = \sqrt{\cosh^2(\phi) - 1}$, $Q'(X) = 0$ yields

$$\begin{aligned} \sqrt{d-1}\cdot k\sqrt{\cosh^2(\phi)-1} &= 2\left(\sqrt{d-1}(2-k)\frac{d}{2\sqrt{d-1}} - (d-k)\right)(d-1)^{-l} \\ &\quad +(d-k) - \sqrt{d-1}(2-k)\cosh(\phi) - o(1)(d-1)^{-l}. \end{aligned}$$

Squaring both sides leads to

$$\begin{aligned} (d-1)k^2(\cosh^2(\phi)-1) &= ((d-k) - \sqrt{d-1}(2-k)\cosh(\phi))^2 \\ &\quad +2((d-k) - \sqrt{d-1}(2-k)\cosh(\phi)) \\ &\quad \cdot(((2-k)d/2) - (d-k))\cdot 2(d-1)^{-l} - o(1)(d-1)^{-l}. \end{aligned}$$

Hence,

$$\begin{aligned} 0 &= 4(d-1)(k-1)\cosh^2(\phi) \\ &\quad -2\sqrt{d-1}(k-2)((d-k) - k(d-2)(d-1)^{-l})\cosh(\phi) \\ &\quad -(d(d-2k+k^2) - 2(d-k)k(d-2)(d-1)^{-l}) - o(1)(d-1)^{-l}. \end{aligned}$$

Calculating $\cosh(\phi)$, we get

$$\cosh(\phi) = \frac{d}{2\sqrt{d-1}} - \frac{k(d-2)^2}{2\sqrt{d-1}(d+k-2)}\cdot\frac{1}{(d-1)^l} + o(1)\frac{1}{(d-1)^l}.$$

Taking into account that $\lambda_i = d - 2\sqrt{d-1}\cdot\cosh(\phi)$ for any $i \in \{2,\ldots,k\}$, we obtain the theorem. □

Note that $T_{d,l,k}$ belongs to the graph class $LS(g, k-1, k)$ with $g(j) = (d-1)^j$. Let $n = k\frac{(d-1)^l - 1}{d-2}$ denote the number of vertices of $T_{d,l,k}$. Using Theorem 3, we obtain that

$$\sum_{i=2}^{k} \lambda_i \leq (k-1)k\frac{(d-2)k^2}{(k(d-2)+2)n(1+o(1))}.$$

There is a factor of $\frac{kd-2k+k^2}{kd-2k+2}$, in which the upper bound differs from the sum of the first k eigenvalues of $T_{d,l,k}$. This factor grows with an increasing number of partitions. However, if d is much larger than k, then this factor is nearly 1.

3.4 Bounds on the k-Section Width of Cartesian Powers of Dense Regular Graphs

In the previous sections we have seen that there are techniques to improve the existing spectral lower bound on the k-partitioning size if the graph has a certain level structure. In this section, we deal with some graphs, which do not have this structure and are described by Cartesian powers of some dense regular graphs. In order to derive new bounds, we define first the edge isoperimetric boundary of a subset of vertices of a graph. Next, we present a new method that can be used to compute bounds on the k-partitioning size of some specific graphs. We conclude by comparing the bounds calculated in this section with the bound techniques described in Section 1.2.

3.4.1 Auxiliary Results

Let $G = (V, E)$ be a graph, $|V| = n$ and let $U \subseteq V$ be a subset of vertices of G. We denote with

$$\partial_G(U) = \{\{u, v\} \in E \mid u \in U,\ v \in \overline{U}\}$$

the *edge isoperimetric boundary* of U. Furthermore,

$$\partial_G(m) = \min_{\substack{U \subseteq V \\ |U|=m}} |\partial_G(U)|.$$

A set $U \subseteq V$ is optimal w.r.t the edge isoperimetric boundary if $\partial_G(|U|) = |\partial_G(U)|$. Denoting with ∇ the optimal k-section width of a graph $G = (V, E)$, we can state the following lemma.

Lemma 10: *Let $V = V_1 \uplus V_2 \uplus \ldots \uplus V_k$ be a balanced k-partitioning of G. Then*

$$\nabla \geq \frac{k}{2} \min\left\{\partial_G\left(\left\lfloor\frac{|V|}{k}\right\rfloor\right), \partial_G\left(\left\lceil\frac{|V|}{k}\right\rceil\right)\right\}.$$

Proof: For $i \neq j$ denote $E_{i,j} = |\{\{u, v\} \in E \mid u \in V_i,\ v \in V_j\}|$ and put $E_{i,i} = 0$, for all $i \in \{1, \ldots, k\}$. Considering $\partial_g(V_i)$ we get

$$\sum_{j=1}^{k} E_{i,j} = |\partial_g(V_i)| \geq \partial_g(|V_i|).$$

Now summarizing this for $i \in \{1, \ldots, k\}$ we obtain

$$\sum_{i=1}^{k}\sum_{j=1}^{k} E_{i,j} = 2\nabla \geq \sum_{i=1}^{k} \partial_g(|V_i|).$$

The lemma follows by taking into account that $||V_i| - |V_j|| \leq 1$ for any $i, j \in \{1, \ldots, k\}$. □

For the following, we define the Cartesian product of two graphs.

Definition 6: *Let $G' = (V', E')$ and $G'' = (V'', E'')$ be two graphs. The* Cartesian product *of $G' \times G''$ is the graph*

$$\begin{aligned} G &= (V, E) \text{ with } V = V' \times V'' \text{ and} \\ E &= \{\{(u', u''), (v', v'')\} \mid \text{with } u' = v' \text{ and } \{u'', v''\} \in E'' \text{ or } u'' = v'' \text{ and } \{u', v'\} \in E'\} \end{aligned}$$

The second Cartesian power of a graph G is obtained by applying $G \times G$ and is denoted by G^2. The graph $G^m = G \times \cdots \times G$ is called the m^{th} *Cartesian power* of G.

In this section, we consider the Cartesian powers of the following graphs. Let $H_{a,l} = (V_{H_{a,l}}, E_{H_{a,l}})$ be constructed from K_a, a even, by partitioning its vertex set into two parts V_l and V_r with $|V_l| = |V_r| = a/2$ and removing l disjoint perfect matchings between these sets, where $l \le \lfloor \frac{a}{4} \rfloor$. Formally,

$$\begin{aligned} V_{H_{a,l}} &= V_l \cup V_r, \quad V_l = \{0, 1, \ldots, a/2 - 1\},\ V_r = \{a/2, \ldots, a-1\}, \\ E_{H_{a,l}} &= \{(u,v) \mid u, v \in V_l, \text{ or } u, v \in V_r\} \cup \\ &\quad \{(u,v) \mid u \in V_l,\ v \in V_r,\ v = a/2 + (u \pm j) \bmod a/2,\ \}, \end{aligned}$$

where $j = 1, \ldots, (a/2 - l)/2$ if $a/2 - l$ is even and $j = 0, \ldots, (a/2 - l - 1)/2$ if $a/2 - l$ is odd. An example of $H_{6,1}$ and $H_{8,2}$ is shown in Figure 3.6(a) and (b). Figure 3.6(c) represents the Cartesian product of $H_{6,1}$ with itself. Here, we presented all incident edges only for the vertex $(2, 2)$.

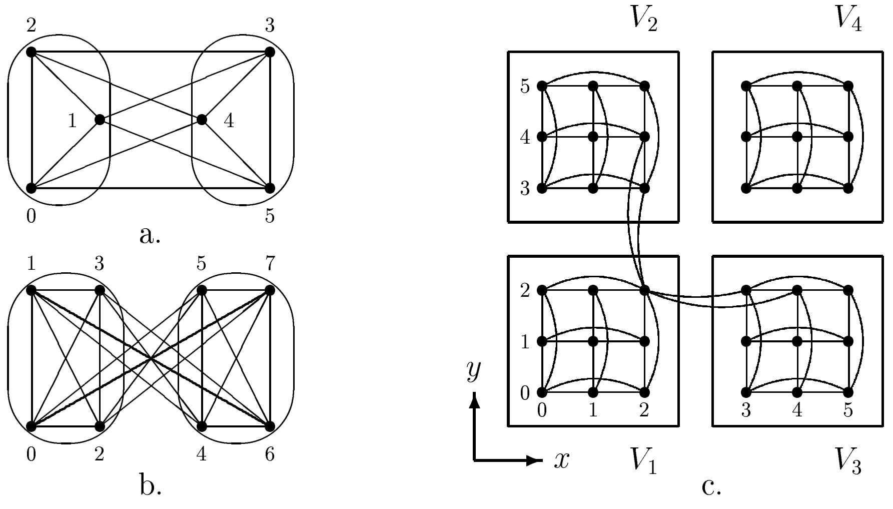


Fig. 3.6: Representations of $H_{6,1}$ (a), $H_{8,2}$ (b), and $H^2_{6,1}$ (c)

We introduce the *lexicographic number* $l(u)$ of a vertex $u = (u^1, \ldots, u^m)$ of the m^{th} Cartesian power of $H_{a,l}$, defined by $l(u) = \sum_{i=1}^{m} u^i a^{n-i}$. Denote

$$L_i^m = \{u \in V_{H_{a,l}} \mid 0 \le l(u) < i\}.$$

We say that two subsets U_1, $U_2 \subset V$ are congruent if U_2 is the image of U_1 in some authomorphism of G. We call a subset of vertices $F \subset V_{H_{a,l}^m}$ a face of $H_{a,l}^m$ of dimension p $(0 \le p \le m)$ if the subset F and $L_{a^p}^m$ are congruent. In [BE01] we have shown the following lemma

Lemma 11: *L_i^m is an optimal set in $H_{a,l}^m$ for any $m \in I\!N$, $l \le \lfloor a/4 \rfloor$ and $i \in \{1, \ldots, a^m\}$.*

This implies

Corollary 1: *Let $V = V_1 \uplus \ldots V_k$ be a balanced k-partitioning of the graph $G = (V, E)$. If each subset V_i, $1 \le i \le k$ is optimal, then the size of this k-partitioning is minimal.*

From this, we obtain the following corollaries.

Corollary 2: *Let k divide a. Then, the optimal k-partitioning size of $H_{a,l}^m$, $l \le \lfloor a/4 \rfloor$, has the form*

$$\nabla = \frac{a(k-1) - kl}{2k} a^m.$$

Corollary 3: *Let p be a constant with $0 < p \le m$. The optimal a^p-partitioning size of $H_{a,l}^m$, $l \le \lfloor a/4 \rfloor$ has the form*

$$\nabla = \frac{(a-l-1)p}{2} a^m.$$

In order to show this, just partition the vertices of $H_{a,l}^m$ into a^p faces of dimension $m - p$.

Since we are only interested in the asymptotic of the k-partitioning size, it is convenient to operate with partitions, where each set is maybe not optimal, but is, in a sense, close to an optimal one. In other words, let $U \subset V_{H_{a,l}^m}$ and let c be some constant. We say that U is quasioptimal if there exists an optimal subset $U' \subset V_{H_{a,l}^m}$, such that $|U \Delta U'| \le c$, where Δ denotes the symmetric difference.

Obviously, for each quasioptimal set $U \subset V_{H_{a,l}^m}$ there exists a c such that $|\partial_{H_{a,l}^m}(U)| - \partial_{H_{a,l}^m}(|U|) \le cam$. Accordingly, we call a balanced partition $V_{H_{a,l}^m} = V_1 \uplus \ldots \uplus V_k$ quasiminimal if there exists a constant c' such that

$$|\nabla - \frac{k}{2} \partial_{H_{a,l}^m}(\lfloor V_{H_{a,l}^m}/k \rfloor)| \le c'm.$$

Here, c' depends on k, c and a, but not on m.

Corollary 4: *Let k be fixed and $V_{H^m_{a,l}} = V_1 \uplus \ldots \uplus V_k$ be a balanced k-partitioning of $H^m_{a,l}$, $l \leq \lfloor a/4 \rfloor$. If each subset V_i is quasioptimal, then the partition is quasiminimal.*

Indeed, similary as in the proof of Lemma 10 we obtain

$$\begin{aligned} 2\nabla &= \sum_{i=1}^{k} |\partial_{H^m_{a,l}}(V_i)| \leq \sum_{i=1}^{k} (\partial_{H^m_{a,l}}(|V_i|) + cam) \\ &\leq k(\partial_{H^m_{a,l}}(\lfloor V_{H^m_{a,l}}/k \rfloor + am) + akmc' = k(\partial_{H^m_{a,l}}(\lfloor V_{H^m_{a,l}}/k \rfloor) + akm(c'+1) \end{aligned}$$

Obviously, $\partial_{H^m_{a,l}}(\lfloor V_{H^m_{a,l}}/k \rfloor)$ is exponential in m if k and a are fixed. Therefore, $m \to \infty$ yields

$$\nabla \approx \frac{k}{2} \partial_{H^m_{a,l}}(\lfloor V_{H^m_{a,l}}/k \rfloor)$$

if the vertices of $H^m_{a,l}$ can be partitioned into k quasioptimal subsets.

In this section we computed exact values for the k-partitioning size of $H^m_{a,l}$ only for "trivial" values of k (see corollaries 2 and 3). Now we give an example, where we are able to compute an asymptotically exact value for the k-section width of $H^m_{a,l}$ for more complicated values of k. For this, consider the following lemma.

Lemma 12: *Let $H^{m-q}_{a,l}$, $l \leq \lfloor a/4 \rfloor$ be partitioned into k quasioptimal subsets. Then, for the constant q and $m \to \infty$, it holds that*

$$\nabla_m \approx \frac{q(a-l-1)}{2} a^m + a^q \nabla_{m-q}.$$

Here we denoted with ∇_m the ka^q-section width of $H^m_{a,l}$ and with ∇_{m-q} the k-section width of $H^{m-q}_{a,l}$.

Proof: Given a partition of $H^{m-q}_{a,l}$ into k quasioptimal subsets, we construct a partition of $H^m_{a,l}$ into ka^q quasioptimnal subsets. For that, we first partition the vertices of $H^m_{a,l}$ into a^q faces of dimension a^{m-q}. Due to corollary 3, the corresponding cut is of size $\frac{q(a-l-1)}{2} a^m$. Now, partition each face into k quasioptimal subsets, assuming that the partitions of all the faces are isomorphic. It remains to note that if a set is quasioptimal in $H^{m-q}_{a,l}$, then it is also quasioptimal in $H^m_{a,l}$. □

Theorem 10: *There exists a partition of $H^m_{a,l}$ into $k = a^p + 1$ quasioptimal subsets for all $m \in I\!N$ and $l \leq \lfloor a/4 \rfloor$.*

Proof: Let $m \geq 2a+1$. Consider first a partition of $H^m_{a,l}$ into $k-1 = a^p$ faces $F_1, \dots, F_{k-1}$ of dimension $m-p$. Now, for $H^m_{a,l}$, let $U_i \subseteq F_i$, $1 \leq i \leq k-1$ with U_i congruent to L^{m-p}_b and $b = \lfloor a^{m-p}/k \rfloor$. We construct a k-partitioning $V = V_1 \uplus \dots \uplus V_k$ with $V_i = F_i \setminus U_i$ for $i \leq k-1$ and $V_k = U_1 \cup \dots \cup U_{k-1}$. Since a and p are constants, we can reassign a constant number of vertices (depending only on a and p) between the parts V_k and V_i with $i \leq k-1$, so that the resulting subsets will be quasioptimal. □

In order to compute the k-section width $\nabla_{m,k}$ of $H^m_{a,l}$ for $k = a^p + 1$, we first compute $g_m = \partial_{H^m_{a,l}}(b)$ for $b = \lfloor a^m/k \rfloor$. Then, g_m satisfies the recursion

$$g_m = g_{m-p} + \frac{p(a-l-1)}{k} a^m + O(m).$$

This provides

$$g_m \approx \frac{(a-l-1)(k-1)p}{k(k-2)} a^m$$

as $m \to \infty$. Hence, we get

$$\nabla_{m,k} \approx \frac{(a-l-1)pa^p}{2(a^p-1)} a^m.$$

Representing $a^p + a^q = a^q(a^{p-q} - 1)$ and applying Theorem 10 and Lemma 12, we get the following corollary

Corollary 5: *Let $p > q \leq 0$. If $m \to \infty$, then the $(a^p + a^q)$-section width ∇ of $H^m_{a,l}$ has the form*

$$\nabla \approx \frac{pa^p - qa^q}{a^p - a^q} \cdot \frac{a-l-1}{2} a^m.$$

3.4.2 Bounds Concerning the k-Section Width of $H^m_{a,l}$

The main theorem of this section is formulated as follows.

Theorem 11: *Let $m > 2$, $a \geq 2$ and $p \in I\!N$ such that $a^{p-1} < k < a^p$. Moreover, we assume that $m > 2(p-1)$. If ∇ denotes the k-section width of $H^m_{a,l}$, $l \leq \lfloor a/4 \rfloor$, then it holds that*

$$\frac{(a-l-1)(p-1)}{2} a^m \leq \nabla \leq (\frac{3}{2}p + 1)(a-l-1)a^m.$$

Proof: To prove the lower bound, we apply Lemma 10 and estimate the minimum. Let $b = \lfloor a^m/k \rfloor$ and let $V_{H^m_{a,l}}$ be partitioned in faces of dimension $m-p+1$. Now L^m_b and L^m_{b+1} are proper subsets of one of such a face, say F. The inner edge boundary for L^m_{b+1} is denoted by $I_{L^m_{b+1}}$ and consists of the edges with one endpoint in L^m_{b+1} and the other endpoint in $F \setminus L^m_{b+1}$. The outer edge boundary is denoted by $O_{L^m_{b+1}}$ and contains the edges with one endpoint in L^m_{b+1} and the other endpoint in $V_{H^m_{a,l}} \setminus F$. The same can also be defined for L^m_b. Then,

$$\begin{aligned}\min\{\partial_{H^m_{a,l}}(b), \partial_{H^m_{a,l}}(b+1)\} &\geq \min\{|O_{L^m_{b+1}}|, |O_{L^m_b}|\} + \min\{|I_{L^m_{b+1}}|, |I_{L^m_b}|\} \\ &\geq b(a-l-1)(p-1) + (m-p-1)(a-l-1)\end{aligned}$$

Since $b \geq a^m/k - 1$, we get

$$\begin{aligned}\nabla &\geq \frac{k}{2}(b(a-l-1)(p-1) + (m-p-1)(a-l-1)) \\ &\geq \frac{k}{2}\frac{a^m}{k}(p-1)(a-l-1)\end{aligned}$$

and the lower bound follows.

To show the upper bound, we first partition $V_{H^m_{a,l}}$ into the faces $F_1, \dots F_a$ of dimension $m-1$ and then, isomorphically partition each F_i into k optimal balanced subsets, $V^i_1, \dots, V^i_k$, $1 \leq i \leq a$. By setting $V_j = \cup^a_{i=1} V^i_j$ for $j \in \{1, \dots, k\}$, we get a partition $V_{H^m_{a,l}} = V_1 \uplus \dots \uplus V_k$. Since k is not a power of a, $|V^i_j| \in \{b, b+1\}$ with $b = \lfloor a^{n-1}/k \rfloor$. Hence, $|V_j| \in \{ab, a(b+1)\}$.

Such a partition can be balanced by deleting vertices from larger parts and adding them to smaller parts. We observe that at most $ak/2$ vertices of the initial partitioning will be reassigned. Therefore, the balancing results in increasing the size of the partition by at most $\frac{a(a-l-1)km}{2}$ and

$$\nabla \leq a\nabla_{a^{m-1}} + \frac{a(a-l-1)km}{2}.$$

Here, we denoted with $\nabla_{a^{m-1}}$ the optimal k-partitioning size of the graph $H^{m-1}_{a,l}$. For $r \geq p$ it follows that:

$$\nabla \leq a^{m-r}\nabla_{a^r} + \frac{a(a-l-1)k}{2}(m + (m-1)a + \dots + (r+1)a^{m-r-1}). \tag{3.9}$$

The recursion 3.9 implies

$$\nabla \leq a^{m-r}\nabla_{a^r} + \frac{a(a-l-1)k}{4a^r}\frac{(ra+a-r)}{(a-1)^2}a^m.$$

Applying this inequality with $r = p$ and taking into account that at least $a^p - k$ parts consist of two vertices, we get

$$\nabla_{a^r} \leq \frac{(a-l-1)p}{2}a^p - (a^p - k) = \frac{(a-l-1)p-2}{2}a^p + k.$$

Note that since $k/a^p < 1$ and $(a/(a-1))^2 \leq 4$ we obtain

$$\nabla \leq a^m \left(\frac{(a-l-1)p-2}{2} a^p + \frac{k}{a^p} + \frac{k}{a^p} \frac{a(a-l-1)}{4} \frac{pa+a-p}{(a-1)^2} \right)$$

and the upper bound follows. □

In [BE01], the authors have shown that every lower bound on the k-section width of the graph $H_{a,l}^m$ provide a lower bound for the m^{th} Cartesian power of any regular graph containing $2a$ vertices and having degree $a-l-1$. Hence, we can state the following corollary.

Corollary 6: *Let $G = (V, E)$ be a regular graph with $|V| = a$, a even and let the degree of G be $a-l-1$, where $l \leq \lfloor a/4 \rfloor$. Let G^m be the m^{th} Cartesian power of this graph. If ∇ denotes the k-section width of G^m, then*

$$\nabla \geq \frac{(a-l-1)a^m \lfloor log_a(k) \rfloor}{2}.$$

3.4.3 The Eigenvalues of $H_{a,l}$

Now, we concentrate on the eigenvalues of the graph $H_{a,l}$ and show that if $k > a$, the lower bounds computed in the previous subsection provide better results than the classical spectral lower bound. Note that for an optimal k-section of this graph the vertices are incident to at least one cut edge and therefore the spectral bounds computed in section 3.2 can not be applied. In order to compute the eigenvalues of $H_{a,l}^m$, we first compute the Laplacian spectrum of the graph $H_{a,l}$.

Denoting with $\lambda_i(G')$ and $\lambda_j(G'')$ the eigenvalues of the Laplacians of the graphs G' and G'', it is known that $\lambda_i(G') + \lambda_j(G'')$ is an eigenvalue of the Cartesian product $G' \times G''$ for any i, j, $1 \leq i, j \leq n$ [CDS95]. Here, n represents the cardinality of both, G' and G''. Therefore, computing the eigenvalues of $H_{a,l}$, we also obtain the spectrum of the m^{th} Cartesian power of this graph.

Theorem 12: *The spectrum of the Laplacian of $H_{a,l}$ has the form*

$$\begin{aligned} S_{H_{a,l}} &= \left\{ \frac{a}{2} - l \pm \left(\frac{a}{2} - l \right) \right\} \\ &\cup \left\{ a - l \pm \frac{\sin\left(\frac{l}{a/2}\pi i\right)}{\sin\left(\frac{1}{a/2}\pi i\right)} \mid i \in \{1, \ldots, a/2-1\} \right\} \end{aligned}$$

Proof: The Laplacian of $H_{a,l}$ has the form $(a-l-1)I_a - B_a$, where I_a is the identity matrix of size a and B_a represents the adjacency matrix of $H_{a,l}$. Since the graph is regular, the Laplacian spectrum can be directly obtained from the eigenvalues of the adjacency matrix. The matrix B_a has the form

$$\begin{pmatrix} M_1 & M_2 \\ M_2 & M_1 \end{pmatrix},$$

where M_1 is the adjacency matrix of a complete graph of size $a/2$ and M_2 is a circulant matrix described as follows. If $a/2-l$ is an odd number, then the first $\frac{a/2-l+1}{2}$ and the last $\frac{a/2-l-1}{2}$ entries in the first row of M_2 equal 1. All other entries of the first row are 0. Since M_2 is a circulant matrix, the entries of the other rows can be derived from the first row. If $a/2-l$ is an even number, then the first entry of the first row equals 0, while the next $\frac{a/2-l}{2}$ and the last $\frac{a/2-l}{2}$ entries equal 1. Similar to the odd case, all other entries of the first row equal 0 and since M_2 is a circulant matrix, the entries of all other rows follow.

In this proof we consider only the odd case. If $a/2-l$ is even, then the same technique can be used. As pointed out in Lemma 5, the spectrum of B_a is the set of the eigenvalues of the matrices M_1+M_2 and M_1-M_2. Both are circulant matrices and the eigenvalues have the form

$$\nu_i^{\pm} = \sum_{j=1}^{a/2-1} (\sqrt[a/2]{1})^{ij} \pm \left(\sum_{j=0}^{\frac{a/2-l-1}{2}} (\sqrt[a/2]{1})^{ij} + \sum_{j=a/2-\frac{a/2-l-1}{2}}^{a/2-1} (\sqrt[a/2]{1})^{ij} \right), \tag{3.10}$$

$0 \le i \le a/2-1$ [Dav79]. For the following, we consider the case $i \neq 0$. Since $\sum_{j=0}^{a/2-1}(\sqrt[a/2]{1})^{ij} = 0$, it holds

$$\sum_{j=0}^{\frac{a/2-l-1}{2}} (\sqrt[a/2]{1})^{ij} + \sum_{j=a/2-\frac{a/2-l-1}{2}}^{a/2-1} (\sqrt[a/2]{1})^{ij} = -\sum_{j=\frac{a/2-l+1}{2}}^{a/2-\frac{a/2-l+1}{2}} (\sqrt[a/2]{1})^{ij}.$$

Then, equation (3.10) results in

$$\nu_i^{\pm} = \sum_{j=1}^{a/2-1} (\sqrt[a/2]{1})^{ij} \mp \left(\sum_{j=\frac{a/2-l+1}{2}}^{a/2-\frac{a/2-l+1}{2}} (\sqrt[a/2]{1})^{ij} \right),$$

which leads to

$$\nu_i^{\pm} = \sum_{j=1}^{a/2-1} (\sqrt[a/2]{1})^{ij} \mp \left(\frac{(\sqrt[a/2]{1})^{i(a/2-\frac{a/2-l-1}{2})} - (\sqrt[a/2]{1})^{i(\frac{a/2-l+1}{2})}}{(\sqrt[a/2]{1})^{i}-1} \right).$$

Taking into account that $(\sqrt[a/2]{1})^i = \cos(\frac{2\pi i}{a/2}) + I\sin(\frac{2\pi i}{a/2})$, we get

$$\nu_i^{\pm} = \sum_{j=1}^{a/2-1} (\sqrt[a/2]{1})^{ij} \mp \nu',$$

where

$$\nu' = \frac{\cos(\frac{2\pi i(\frac{a/2+l+1}{2})}{a/2}) - \cos(\frac{2\pi i(\frac{a/2-l+1}{2})}{a/2}) + I\left(\sin(\frac{2\pi i(\frac{a/2+l+1}{2})}{a/2}) - \sin(\frac{2\pi i(\frac{a/2-l+1}{2})}{a/2})\right)}{\cos(\frac{2\pi i}{a/2}) - 1 + I\sin(\frac{2\pi i}{a/2})}$$

and after some calculations this implies

$$\nu_i^{\pm} = -1 \mp \frac{\sin\left(\frac{l}{a/2}\pi i\right)}{\sin\left(\frac{1}{a/2}\pi i\right)},$$

where $i \in \{1, \ldots, a/2-1\}$. Since the eigenvalues of the Laplacian have the form $a - l - 1 + \nu_i^{\pm}$, we obtain the theorem.

Let us now analyze the case $i = 0$. Then

$$\sum_{j=0}^{\frac{a/2-l-1}{2}} (\sqrt[a/2]{1})^{ij} + \sum_{j=a/2-\frac{a/2-l-1}{2}}^{a/2-1} (\sqrt[a/2]{1})^{ij} = a/2 - l$$

and

$$\sum_{j=1}^{a/2-1} (\sqrt[a/2]{1})^{ij} = a/2 - 1.$$

This implies that $\nu_0^{\pm} = a/2 - 1 \pm (a/2 - l)$ and the theorem follows. □

We consider the case when $k < m$ and $k > a$ and compare the classical spectral lower bound with the bounds calculated in this section. Obviously, the first $k-1$ non-zero eigenvalues of $H_{a,l}^m$ are equal to the second smallest eigenvalue of the graph $H_{a,l}$. Therefore, to compare the bounds mentioned before with each other, we are only interested in the second smallest eigenvalue of $H_{a,l}$.

Corollary 7: *The second smallest eigenvalue of $H_{a,l}$ equals $a - 2l$.*

Proof: We have to show that

$$a - l \pm \frac{\sin\left(\frac{l}{a/2}\pi i\right)}{\sin\left(\frac{1}{a/2}\pi i\right)} \geq a - 2l \tag{3.11}$$

for any $i \in \{1, \ldots, a/2\}$. Equation (3.11) is equivalent to

$$\left|\frac{\sin\left(\frac{l}{a/2}\pi i\right)}{\sin\left(\frac{1}{a/2}\pi i\right)}\right| \leq l.$$

Using induction on l, it can be proved that

$$\left|\frac{\sin(l\phi)}{\sin(\phi)}\right| \leq l$$

for any ϕ. For $l = 1$ equality holds. Assume that the inequality is true for $l - 1$. Then

$$|\sin((l-1)\phi)| \leq (l-1)|\sin(\phi)|$$

and

$$\begin{aligned} |\sin(l\phi)| &\leq |\sin((l-1)\phi)\cos(\phi)| + |\cos((l-1)\phi)\sin(\phi)| \\ &\leq (l-1)|\sin(\phi)| + |\sin(\phi)| \leq l|\sin(\phi)| \end{aligned}$$

Setting $\phi = \frac{1}{a/2}\pi i$, we obtain the corollary. □

Denoting by ∇ the k-section width of the graph $H^m_{a,l}$, where $k < m$ and $l \leq \lfloor a/4 \rfloor$, the classical spectral lower bound results in the expression (even if $k|a^m$ does not hold)

$$\nabla \geq \left\lfloor \frac{a^m}{k} \right\rfloor \frac{(a-2l)(k-1)}{2},$$

while the lower bound computed in this section leads to

$$\nabla \geq \frac{a^m(a-l-1)\lfloor \log_a(k) \rfloor}{2}.$$

For reasonable large values of k, it holds

$$\left\lfloor \frac{a^m}{k} \right\rfloor \frac{(a-2l)(k-1)}{2} \leq \frac{a^m(a-l-1)\lfloor \log_a(k) \rfloor}{2}.$$

Another lower bound technique consists of the embedding method of Leighton [Lei92]. It is known that for the bisection width bw of a graph $G = (V, E)$ the following lower bound holds:

$$bw \geq \frac{n^2}{4 \cdot cong}, \tag{3.12}$$

where $cong$ is the maximal congestion of an embedding of G into the complete graph of cardinality $n = |V|$. Inequality (3.12) can be also generalized for the k-section width of a graph as follows.

Theorem 13: *Let $G = (V, E)$ be a graph with $n = |V|$ vertices and assume that $k|n$. For the k-partitioning size ∇ of G, it holds that*

$$\nabla \geq \frac{n^2(k-1)}{2k \cdot cong},$$

where cong denotes the maximal congestion of an embedding of G into the complete graph of cardinality n.

Proof: The main idea is based on the fact that we can connect every pair of nodes in G with a path so that no edge of G is contained in more than $cong$ paths. We will then argue that if G has a k-section containing fewer than $\frac{n^2(k-1)}{2k \cdot cong}$ edges, then the complete graph will have a k-section with fewer than $\frac{n^2(k-1)}{2k}$ edges. The proof is concluded by showing that this is not possible.

For the purposes of contradiction, assume that G has a k-section with $\nabla < \frac{n^2(k-1)}{2k \cdot cong}$ edges. This means that there is a partition of the vertices of G into equal sides so that at most ∇ edges connect the sides to each other. The same partition induces also a k-section of the complete graph. Moreover, any edge connecting one side of the complete graph to another must correspond to a path in G containing an edge of the k-section. Since at most $cong$ paths can contain any edge of G, this means that the complete graph has a k-section with fewer than $cong \frac{n^2(k-1)}{2k \cdot cong} = \frac{n^2(k-1)}{2k}$ edges.

However, any k-section of the complete graph must have at least $\frac{n^2(k-1)}{2k}$ edges. This is because each of the n/k vertices of one partition are connected to each of the n/k vertices of any other partition. The contradiction implies that $\nabla \geq \frac{n^2(k-1)}{2k \cdot cong}$. □

In the following, we show that the maximal congestion of any embedding of $H^m_{a,l}$ in the complete graph of cardinality a^m satisfies the inequality

$$cong \geq \frac{a^m}{a - 2l}.$$

In order to prove the inequality described above, partition the vertices of $H_{a,l}^m$ among their lexicographic ordering into two parts. The vertices with a lexicographic number at most $\frac{a^m}{2} - 1$ are in one partition and the others are in the other partition. There are $\frac{a^m}{2}(\frac{a}{2} - l)$ edges between these two parts of the partition. Connecting any pair of vertices of $H_{a,l}^m$ with a path, one edge will contain at least $\left(\frac{a^m}{2} \cdot \frac{a^m}{2}\right) / \left(\frac{a^m}{2}(\frac{a}{2} - l)\right)$ paths and the inequality follows.

Using Theorem 13 the k-section width of $H_{a,l}^m$ is bounded by (even if $k|a^m$ does not hold)

$$\nabla \geq \frac{a-2l}{2} \cdot \left(\left(a^m - \left\lfloor \frac{a^m}{k} \right\rfloor\right) - \frac{1}{a^m} \left\lceil \frac{a^m}{k} \right\rceil \left(\left\lceil \frac{a^m}{k} \right\rceil k - a^m\right)\right).$$

The inequality above can be rewritten as

$$\nabla \geq \frac{a^m(a-2l)}{2} \cdot \left(\left(1 - \frac{1}{a^m} \left\lfloor \frac{a^m}{k} \right\rfloor\right) - O\left(\frac{1}{a^m}\right)\right).$$

For large values of m and some large but reasonable values of k, it is easy to see that the lower bound computed in Theorem 11 is better than the bound computed by the embedding method.

3.5 Summary

Let us summarize the results of this chapter. In Section 3.1, we analyzed the spectral lower bound described in (1.1) and could show that this bound is tight only for a few graphs. In Section 3.2, we derived new relations between structural and spectral properties of graphs. Using these new relations, we obtained improved spectral lower bounds on the k-section width of level structured graphs. If the growth function $g(j)$, which describes the connectivity between consecutive levels containing vertices of distance j and vertices of distance $j+1$ from the cut, increases greatly, then we improved the old spectral lower bound by a constant factor. Otherwise, we showed that the lower bound depends on $(\sum_{i=1}^k \lambda_i)^{\frac{\alpha+1}{2}}$, $0 \leq \alpha < 1$, instead of $\sum_{i=1}^k \lambda_i$. In Section 3.3, we presented some graph classes, for which the new bounds are asymptotically tight. In Section 3.4, we described a new technique to derive bounds on the k-section width of Cartesian powers of dense regular graphs.

4. ALTERNATING DIRECTION AND OPTIMAL LOAD BALANCING SCHEMES

In this chapter, we improve the existing load balancing algorithms on certain graphs, by using their structural and spectral properties. In Section 4.1, we consider Cartesian products of graphs and derive a new load balancing scheme for them. In Section 4.2, we describe a simple optimal scheme. In Section 4.3, we discuss experimental results concerning the load balancing algorithms introduced in this chapter. In the last section, we present scalable and non-scalable network topologies having a small vertex degree and only $O(\log^k(n))$ distinct eigenvalues (n denotes the cardinality of the graphs and $k \in I\!N$ is a constant). These topologies are well-suited for load balancing applications based on the optimal scheme.

4.1 The Alternating Direction Iterative Scheme

In [XL97], Xu and Lau proposed a new algorithm for Cartesian products of two paths resp. two cycles. They showed that if we apply in each iteration an FOS step, first along one component of the product, and second along the other component of the product, then the number of iteration steps needed to balance the load is reduced significantly. However, they did not consider faster diffusive schemes, such as SOS, and did not analyze the quality of the resulting flow. Here, we generalize this method, called the *alternating direction iterative scheme*, to Cartesian products of arbitrary graphs. We show that one can reduce the number of load balance iterations by alternating the algorithm from one component of the product to the other and analyze the resulting flow. Moreover, we will propose variants of this scheme, resulting in a better flow than the classical alternating direction iterative scheme.

4.1.1 The Convergence of the Alternating Direction Iterative Scheme

Let $G = (V, E)$ be the Cartesian product of $G' = (V', E')$ with $G'' = (V'', E'')$. We denote with $\lambda'_1, \ldots, \lambda'_p$ and with $\lambda''_1, \ldots, \lambda''_q$ the eigenvalues of the Laplacians of G' and G'', respectively. Let $p = |V'|$ and $q = |V''|$. As described in the previous chapter, for any eigenvalue λ of the Laplacian of G there exist some integers $i \in \{1, \ldots, p\}$ and $j \in \{1, \ldots, q\}$ such that

$$\lambda = \lambda'_i + \lambda''_j.$$

We denote with I_k the identity matrix of order k and with L the Laplacian of G. Furthermore, we denote with L' and L'' the Laplacian matrices of G' and G'', respectively. Then, it holds that

$$L = I_q \otimes L' + L'' \otimes I_p.$$

Here, the direct product of two matrices $Q \in I\!R^{p \times p}$ and $R \in I\!R^{q \times q}$ is a square matrix of size pq defined by

$$Q \otimes R = \begin{pmatrix} Q_{1,1}R & Q_{1,2}R & \cdots & Q_{1,p}R \\ \vdots & \vdots & \ddots & \vdots \\ Q_{p,1}R & Q_{p,2}R & \cdots & Q_{p,p}R \end{pmatrix}.$$

Let $z'_i \in I\!R^p$ be an eigenvector of L', $1 \le i \le p$, corresponding to the eigenvalue λ'_i and let $z''_j \in I\!R^q$ be an eigenvector of L'', $1 \le j \le q$, corresponding to the eigenvalue λ''_j. Then, $(i_1 z'^t_i, i_2 z'^t_i, \ldots, i_q z'^t_i)^t$ and $(z''_{i,1}(j_1, \ldots, j_p), \ldots, z''_{i,q}(j_1, \ldots, j_p))^t$ are eigenvectors of $I_q \otimes L'$ and $L'' \otimes I_p$, respectively. Again, we denoted with $z'_{i,l}$ and $z''_{j,l}$ the l^{th} entry of the vectors z'_i and z''_j, respectively, and $i_1, \ldots, i_q$, $j_1, \ldots, j_p$ are arbitrary real numbers. Then, $(z''_{i,1} z'^t_i, \ldots, z''_{i,q} z'^t_i)^t$ is an eigenvector of L corresponding to the eigenvalue $\lambda'_i + \lambda''_j$ and it is obvious that the matrices $I_q \otimes L'$ and $L'' \otimes I_p$ have a common system of eigenvectors.

Now we apply a load balancing strategy on G, which is known to be solving linear systems as the *alternating direction iterative scheme* (ADI) [Var62]. Within an iteration, first balance *along one component* of the Cartesian product (for example G') using FOS, and second *along the other component* (in our case G'') using FOS. The resulting scheme ADI-FOS has the form

$$\begin{aligned} \forall\, e = \{(u'_i, u''_j), (u'_i, v''_l)\} \in E: \; y^{k-1}_{ij,il} &= \alpha''(w^{k-1}_{(u'_i, u''_j)} - w^{k-1}_{(u'_i, v''_l)}); \\ \overline{w}^k_i &= w^{k-1}_i - \sum_{e=\{(u'_i,u''_j),(u'_i,v''_l)\} \in E} y^{k-1}_{ij,il} \text{ and} \end{aligned} \quad (4.1)$$

$$\begin{aligned} \forall\, e = \{(u'_i, u''_j), (v'_l, u''_j)\} \in E: \; y^{k-1}_{ij,lj} &= \alpha'(\overline{w}^k_{(u'_i, u''_j)} - \overline{w}^k_{(v'_l, u''_j)}); \\ w^k_i &= \overline{w}^k_i - \sum_{e=\{(u'_i,u''_j),(v'_l,u''_j)\} \in E} y^{k-1}_{ij,lj} \end{aligned} \quad (4.2)$$

We denoted with $w^k_{(u'_i,u''_j)}$ and $w^k_{(v'_l,u''_j)}$ the load of $(u'_i, u''_j) \in V$ and $(v'_l, u''_j) \in V$, respectively, after the k^{th} iteration. Again α' and α'' has to be chosen, such that $-1 < 1 - \alpha'\lambda'_i < 1$ and $-1 < 1 - \alpha''\lambda''_j < 1$ for any $2 \le i \le p$ and $2 \le j \le q$. Denoting with $M' = 1 - \alpha' L'$ and $M'' = 1 - \alpha'' L''$ the diffusion matrices of G' and G'', equation (4.2) can be rewritten as follows:

$$w^{k+1} = (M'' \otimes I_p)(I_q \otimes M')w^k = (M'' \otimes M')w^k.$$

In the sequel we denote with μ'_i the eigenvalue of M' and with μ''_j the eigenvalue of M'', while μ_{ij} represent the eigenvalues of the diffusion matrix M of G, where $1 \le i \le p$ and $1 \le j \le q$.

Theorem 14: *Let G be the Cartesian product of two connected graphs G' and G'' having the diffusion matrices M' and M''. Furthermore, let γ_M and $\gamma_{M''\otimes M'}$ be the diffusion norms of M and $M'' \otimes M'$, respectively. Then, $\gamma_{M''\otimes M'} < \gamma_M$. If G' and G'' are isomorphic, then $\gamma_{M''\otimes M'} < \gamma_M^2$, i.e. FOS combined with the ADI scheme needs only half the number of iterations to guarantee the same upper bound on the error compared with the FOS.*

Proof: Obviously, the diffusion norm of $M' \otimes M''$ is $\gamma_{M''\otimes M'} = \max\{\gamma_{M'}, \gamma_{M''}\}$. W.l.o.g. we assume that $\gamma_{M'} \ge \gamma_{M''}$. Now, for G it holds that

$$\gamma = 1 - \frac{2\lambda'_2}{\lambda'_p + \lambda''_q + \lambda'_2} = \frac{\lambda'_p + \lambda''_q - \lambda'_2}{\lambda'_p + \lambda''_q + \lambda'_2}.$$

It can be easily shown that

$$\frac{\lambda'_p - \lambda'_2}{\lambda'_p + \lambda'_2} < \frac{\lambda'_p + \lambda''_q - \lambda'_2}{\lambda'_p + \lambda''_q + \lambda'_2}.$$

If G' and G'' are isomorphic, then assume that $\gamma_{M''\otimes M'} \ge \gamma_M^2$. This is equivalent to

$$\frac{\lambda'_p - \lambda'_2}{\lambda'_p + \lambda'_2} \ge \left(\frac{\lambda'_p + \lambda'_p - \lambda'_2}{\lambda'_p + \lambda'_p + \lambda'_2}\right)^2,$$

which implies

$$(\lambda'_p - \lambda'_2)(2\lambda'_p + \lambda'_2)^2 \ge (\lambda'_p + \lambda'_2)(2\lambda'_p - \lambda'_2)^2.$$

Then, we get

$$-\lambda'^3_2 \ge \lambda'^3_2$$

leading to contradiction and the theorem follows. □

In order to obtain a faster convergence of the alternating direction iterative scheme we can also apply SOS instead of FOS among one component of a product. Then, within the first iteration, apply FOS among both components of the Cartesian product (one after another), and in the following iterations, balance alternating along the first component using FOS and along the second component using SOS. The value of $\beta_{M''\otimes M'}$ used by SOS is set to

$$\beta_{M''\otimes M'} = \frac{2}{1+\sqrt{1-\gamma^2_{M''\otimes M'}}},$$

in order to obtain the best convergence rate of this load balancing scheme. M'' and M' denote the diffusion matrices of G'' and G', respectively. Again, we consider the optimal values $\alpha' = \frac{2}{\lambda'_2+\lambda'_p}$ and $\alpha'' = \frac{2}{\lambda''_2+\lambda''_q}$ as described in chapter 2. Then, ADI-SOS has the form

$\forall\, e = \{(u'_i, u''_j), (u'_i, v''_l)\} \in E$:

$$y^{k-1}_{ij,il} = \begin{cases} \alpha''(w^{k-1}_{(u'_i,u''_j)} - w^{k-1}_{(u'_i,v''_l)}) & , \quad \text{if } k = 1,2 \\ (\beta_{M''\otimes M'} - 1)y^{k-3}_{ij,il} + \alpha''(w^{k-1}_{(u'_i,u''_j)} - w^{k-1}_{(u'_i,v''_l)}), & \text{otherwise} \end{cases}$$

$$\overline{w}^k_i = w^{k-1}_i - \sum_{e=\{(u'_i,u''_j),(u'_i,v''_l)\}\in E} y^{k-1}_{ij,il} \qquad \text{and}$$

$\forall\, e = \{(u'_i, u''_j), (v'_l, u''_j)\} \in E$:

$$y^{k-1}_{ij,lj} = \begin{cases} \alpha'(\overline{w}^k_{(u'_i,u''_j)} - \overline{w}^k_{(v'_l,u''_j)}) & , \quad \text{if } k = 1 \\ (\beta_{M''\otimes M'} - 1)y^{k-2}_{ij,lj} + \beta_{M''\otimes M'}\alpha'(\overline{w}^k_{(u'_i,u''_j)} - \overline{w}^k_{(v'_l,u''_j)}), & \text{otherwise} \end{cases}$$

$$w^k_i = \overline{w}^k_i - \sum_{e=\{(u'_i,u''_j),(v'_l,u''_j)\}\in E} y^{k-1}_{ij,lj}$$

or in matrix notation

$$w^1 = (M''\otimes M')w^0 \text{ and } w^k = \beta_{M''\otimes M'}(M''\otimes M')w^{k-1} + (1-\beta_{M''\otimes M'})w^{k-2},\ k = 2,3,\ldots$$

Similar to Theorem 14, we obtain the following result for ADI-SOS.

Theorem 15: *Let G be the Cartesian product of two connected graphs G' and G'' with the diffusion matrices M' and M'' and let γ_M and $\gamma_{M''\otimes M'}$ be the diffusion norms of M and $M''\otimes M'$. Then, ADI-SOS converge faster than SOS, i.e.*

$$(\beta_{M''\otimes M'} - 1)^{k/2}(1 + k\sqrt{1-\gamma^2_{M''\otimes M'}}) \le (\beta_M - 1)^{k/2}(1 + k\sqrt{1-\gamma^2_M}).$$

If G' and G'' are isomorphic, then

$$(\beta_{M''\otimes M'} - 1)^{k/2}(1 + k\sqrt{1-\gamma^2_{M''\otimes M'}}) \le (\beta_M - 1)^{k\sqrt{2}/2}(1 + k\sqrt{2}\sqrt{1-\gamma^2_M}),$$

i.e. ADI-SOS needs $\sqrt{2}$ less iteration steps than SOS in order to guarantee the same upper bound on the error.

Proof: Again, assume w.l.o.g. that $\gamma_{M'} \geq \gamma_{M''}$, which implies $\gamma_{M''\otimes M'} = \gamma_{M'}$. Using $\gamma_{M'} \leq \gamma_M$, the first statement of the theorem can be easily checked by induction on k. For the second statement of the theorem, we use the result of Theorem 14, namely that $\gamma_{M'} < \gamma_M^2$. This implies

$$\left(\frac{1-\sqrt{1-\gamma_{M'}}}{1+\sqrt{1-\gamma_{M'}}}\right)^{\frac{k\sqrt{2}}{2}} (1+k\sqrt{2}\sqrt{1-\gamma_{M'}}) \leq \left(\frac{1-\sqrt{1-\gamma_M^2}}{1+\sqrt{1-\gamma_M^2}}\right)^{\frac{k\sqrt{2}}{2}} (1+k\sqrt{2}\sqrt{1-\gamma_M^2}).$$

By analyzing the function

$$f(X) = \left(\frac{2}{1+\sqrt{1-X}} - 1\right)^{\sqrt{2}} - \left(\frac{2}{1+\sqrt{1-X^2}} - 1\right)$$

we obtain

$$\left(\frac{2}{1+\sqrt{1-\gamma_{M'}}} - 1\right)^{\sqrt{2}} \geq \left(\frac{2}{1+\sqrt{1-\gamma_{M'}^2}} - 1\right).$$

Since $1+k\sqrt{2}\sqrt{1-\gamma_{M'}} \geq 1+k\sqrt{1-\gamma_{M'}^2}$ for any $0 \leq \gamma_{M'} < 1$, the theorem follows. □

Due to the fact that the Chebyshev scheme described in chapter 2 can be regarded to perform asymptotically identical to SOS, we obtain similar results for the Chebyshev scheme combined with ADI. Again, in the first iteration step we use FOS among both components of a Cartesian product. In the following iterations, we alternate one FOS step along the first component with one Chebyshev step along the second component. The parameters β_1, β_2, ..., β_k are described by

$$\beta_1 = 1,\ \beta_2 = \frac{2}{2-\gamma_{M''\otimes M'}^2},\ \beta_k = \frac{4}{4-\gamma_{M''\otimes M'}^2\beta_{k-1}},\ k = 3, 4, \ldots$$

We also analyzed the possibility of alternating fast diffusion schemes such as SOS or Chebyshev among both components of a Cartesian product, but in this case we could not guarantee the convergence of the algorithm. Therefore, we do not consider this kind of alternating algorithms.

4.1.2 The Flow of the Alternating Direction Iterative Scheme

As described in Chapter 2, the flow computed by a polynomial-based diffusion scheme results in an l_2-optimal flow. However, this is not true for the flow computed by the alternating direction iterative scheme presented in the previous section. Denote with z_{ij} the orthogonal (not necessarily normalized) eigenvectors of the matrices $I_q \otimes B_{G'}$ and

$B_{G''} \otimes I_p$ and by A' and A'' the edge-incidence matrices of the graphs G' and G''. Now let $A_1 = I \otimes A'^T$ and $A_2 = A''^T \otimes I$. Let w_1^k be the load after applying k times FOS alternating along both components and additionally, one time only along the first component of the product. Denote with w_2^k the load after k iteration steps, where we applied k times FOS along both components of the product. Moreover, let F_1^{k+1} and F_2^{k+1} be the flows in iteration $k+1$ among the first and the second dimension, which are

$$F_1^{k+1} = \alpha' A_1^T w_2^k \quad \text{and} \quad F_2^{k+1} = \alpha'' A_2^T w_1^{k+1}.$$

For the total flow in these directions holds

$$\begin{aligned} F_2 &= \sum_{k=0}^{\infty} F_2^{k+1} = \alpha'' \sum_{k=0}^{\infty} A_2^T \Big(\sum_{\substack{i \in \{1,\dots,p\} \\ j \in \{1,\dots,q\}}} \mu_i' (\mu_i' \mu_j'')^k z_{ij} \Big) \\ &= \alpha'' \sum_{\substack{i \in \{1,\dots,p\} \\ j \in \{1,\dots,q\}}} \mu_i' \sum_{k=0}^{\infty} (\mu_i' \mu_j'')^k A_2^T z_{ij} = \sum_{\substack{i \in \{1,\dots,p\} \\ j \in \{1,\dots,q\}}} \frac{\alpha''(1-\alpha'\lambda_i')}{\alpha'\lambda_i' + \alpha''\lambda_j'' - \alpha'\alpha''\lambda_i'\lambda_j''} A_2^T z_{ij} \\ F_1 &= \sum_{k=0}^{\infty} F_1^{k+1} = \alpha' \sum_{k=0}^{\infty} A_1^T \Big(\sum_{\substack{i \in \{1,\dots,p\} \\ j \in \{1,\dots,q\}}} (\mu_i' \mu_j'')^k z_{ij} \Big) \\ &= \alpha' \sum_{\substack{i \in \{1,\dots,p\} \\ j \in \{1,\dots,q\}}} \sum_{k=0}^{\infty} (\mu_i' \mu_j'')^k A_1^T z_{ij} = \sum_{\substack{i \in \{1,\dots,p\} \\ j \in \{1,\dots,q\}}} \frac{\alpha'}{\alpha'\lambda_i' + \alpha''\lambda_j'' - \alpha'\alpha''\lambda_i'\lambda_j''} A_1^T z_{ij}. \end{aligned}$$

If the graphs G' and G'' are isomorphic, then $\alpha' = \alpha''$ and F_2 resp. F_1 result in the values

$$\begin{aligned} F_2 &= \sum_{i,j \in \{1,\dots,p\}} \frac{1-\alpha'\lambda_i'}{\lambda_i' + \lambda_j'' - \alpha'\lambda_i'\lambda_j''} A_1^T z_{ij} \\ F_1 &= \sum_{i,j \in \{1,\dots,p\}} \frac{1}{\lambda_i' + \lambda_j'' - \alpha'\lambda_i'\lambda_j''} A_1^T z_{ij}. \end{aligned}$$

We observe that the quality of the flow depends on the value of α' and this is already true if we consider ADI-SOS. The result of the classical diffusion method results in a l_2-minimal flow, in which case we get

$$F = \sum_{k=0}^{\infty} \alpha A^T w^k = \alpha \sum_{l \in \{1,\dots,pq\}} \sum_{k=0}^{\infty} \mu_l^k A^T z_l = \sum_{\substack{i \in \{1,\dots,p\} \\ j \in \{1,\dots,q\}}} \frac{1}{\lambda_i' + \lambda_j''} A^T z_{ij}.$$

Note that M and L have the same eigenvectors z_{ij} corresponding to the eigenvalues μ_{ij} and $\lambda_i' + \lambda_j''$, respectively. Due to this result the l_2-minimal flow of a polynomial-based load balancing scheme does not depend on α, while the flow of the ADI-method only converges to the l_2-minimal flow if α' converges to zero, implying a larger number of iterations.

As an example, consider an $n \times n$ torus with the load $w^0 = K \cdot z_1 + K \cdot z_{n^2}$, where K is a constant, i.e. each node has a load of 0 or $2K$ and each two adjacent nodes have different loads. It is easy to see that there is a balancing flow with a flow of $\frac{K}{4}$ for each edge (independent of n). Since the $n \times n$ torus is the Cartesian product of two cycles of length n, it holds that $\alpha' = \alpha''$. Using the ADI-method, we get $F_1 = K\frac{1}{8-16\alpha'}A_1^T z_{n^2}$ and $F_2 = K\frac{1-4\alpha'}{8-16\alpha'}A_2^T z_{n^2}$, i.e. the flow depends on α'. The optimal value for α' is $\frac{2}{\lambda'_2+\lambda'_n}$ if the number of iterations is to be kept minimal. For the $n \times n$ torus, this optimal value for α' converges to $\frac{1}{2}$ with increasing n. Therefore, the flows F_1 and F_2 for each edge are also increasing with increasing n. It shows that, with this scheme and this initial load, the load is only slightly balanced in each iteration, whereas a large amount of load is added on all circles of length 4, leading to a final flow with many heavy-weighted circles.

To avoid this problem, we construct a new scheme called *mixed direction iterative* (MDI) scheme as follows. In each iteration with even number we first balance *along component 1* and then *along component 2*, whereas in each iteration with odd number we first balance *along component 2* and then *along component 1*, i.e. the order of the components for each sub-iteration are changed for each iteration. Using similar arguments as before, the diffusion norm is the same as in the ADI-method, but we get the flows

$$F_1 = \sum_{i,j\in\{1,\dots,n\}} \frac{1+\mu'_i\mu''^2_j}{1-\mu'^2_i\mu''^2_j}A_1^T z_{ij} \quad \text{and} \quad F_2 = \sum_{i,j\in\{1,\dots,n\}} \frac{\mu'_i+\mu'_i\mu''_j}{1-\mu'^2_i\mu''^2_j}A_2^T z_{ij}.$$

Now, for the same load as before $w^0 = Fz_1 + Fz_{n^2}$, we can observe that, if n increases, the flows F_1 and F_2 are bounded. Although the flow is not necessarily bounded for other initial load distributions, MDI generally calculates a smaller flow than ADI, which we will also see in the experiments.

4.2 Optimal Schemes

In [DFM99], an optimal load balancing scheme OPS was introduced. In this section, we describe a simpler optimal scheme OPT already presented in [EFMP99].

As it was shown in Chapter 2, the error e^k of any polynomial-based scheme applied on a graph G satisfies

$$e^k = p_k(M)(\sum_{i=2}^{n} z_i) = \sum_{i=2}^{n} p_k(M)z_i.$$

Let m be the number of different eigenvalues of the Laplacian L of G. We apply the local iterative algorithm of equation (2.2), which use the distinct non-zero eigenvalues $\tilde{\lambda}_k$,

$2 \le k \le m$, (in any order) as follows:

$$\begin{aligned} \forall\, e = \{v_i, v_j\} \in E:\ y_{i,j}^{k-1} &= \frac{1}{\tilde{\lambda}_k}(w_i^{k-1} - w_j^{k-1}); \quad x_e^k = x_e^{k-1} + \delta_{i,j} y_{i,j}^{k-1}; \\ \text{and} \quad w_i^k &= w_i^{k-1} - \sum_{e=\{v_i,v_j\}\in E} y_{i,j}^{k-1} \end{aligned} \tag{4.3}$$

This means that in each iteration k, a node i adds a flow of $\delta_{i,j}\frac{1}{\tilde{\lambda}_k}(w_i^{k-1} - w_j^{k-1})$ to the flow over edge $\{i,j\}$, choosing a different eigenvalue for each iteration. Again, $\delta_{i,j} = 1$ if $i > j$ and -1 otherwise. Therefore,

$$w^k = \left(I - \frac{1}{\tilde{\lambda}_{k+1}} L\right) w^{k-1},$$

and for error e^{m-1} after $m-1$ iterations we obtain

$$e^{m-1} = \prod_{i=2}^{m} \left(I - \frac{1}{\tilde{\lambda}_i} L\right) \sum_{j=2}^{n} z_j = 0.$$

After $m-1$ iterations we have $e^{m-1} = 0$ and the flow is l_2-minimal as stated in Chapter 2. For each iteration a different eigenvalue of L is used, eliminating all corresponding eigenvectors to this eigenvalue. The order of the eigenvalues may be arbitrary, however, the order may influence the numerical stability of the calculation (see Section 4.3). No further parameters beside the eigenvalues have to be used for this scheme and it has a very simple construction.

As an example, consider the d-dimensional hypercube $(Q(d))$ and consider the initial load $w^0 = (K, 0, \ldots, 0)$. An easy and well-known method for load balancing on the hypercube is based on the dimensional exchange strategy, where in each iteration only the balancing in one dimension is allowed and in each iteration a node equalizes its load with the load of its neighbor. It is easy to see, that the error e^d will be reduced to zero after d steps. The resulting flow in the l_2-norm is

$$F_{de} = K\sqrt{\left(\frac{2^d - 1}{2^{d+1}}\right)}.$$

The d-dimensional hypercube has $d+1$ distinct eigenvalues: $0, 2, 4, \ldots, 2d$. The OPT scheme computes in each iteration k the flow $y_{i,j}^k = \frac{1}{2(k+1)}(w_i^k - w_j^k)$. over edge $e = \{v_i, v_j\}$.

The l_2-minimal flow is

$$F_{l_2} = \sqrt{\sum_{i=0}^{d-1} \left(\frac{K - \frac{K}{2^d}(\sum_{j=0}^{i} \binom{d}{j})}{\binom{d}{i}(d-i)} \right)^2},$$

G	Spectrum of B	Spectrum of L	$\frac{\lambda_2}{\lambda_n}$
K_n	$-1[n-1],\ (n-1)$	$0,\ n[n-1]$	1
$K_{n,n}$	$-n,\ 0[2n-2],\ n$	$0,\ n[2n-2],\ 2n$	$1/2$
P_n	$2 \cdot \cos(\frac{\pi}{n+1}j)$ $j = 1, \dots, n$	$2 - 2 \cdot \cos(\frac{\pi}{n}j)$ $j = 0, \dots, n-1$	$\frac{2-2\cdot\cos(\frac{\pi}{n})}{4}$
C_n	$2 \cdot \cos(\frac{2\pi}{n}j)$ $j = 0, \dots, n-1$	$2 - 2 \cdot \cos(\frac{2\pi}{n}j)$ $j = 0, \dots, n-1$	$\frac{2-2\cdot\cos(\frac{2\pi}{n})}{4}$
S_n	$0,\ \sqrt{n-1}$	$0,\ 1,\ n$	$1/n$
$Q(d)$	$-d,\ -(d-2), \dots, (d-2), d$	$0, 2, 4, \dots, 2d$	$1/d$
G_n	$2 \cdot (\cos(\frac{\pi}{n+1}j_1) + \cos(\frac{\pi}{n+1}j_2))$ $j_1, j_2 = 1, \dots, n$	$4 - 4 \cdot (\cos(\frac{\pi}{n}j_1) + \cos(\frac{\pi}{n}j_2))$ $j_1, j_2 = 0, \dots, n-1$	$\frac{1-\cos(\frac{\pi}{n})}{2}$
T_n	$2 \cdot (\cos(\frac{2\pi}{n}j_1) + \cos(\frac{2\pi}{n}j_2)$ $j_1, j_2 = 0, \dots, n-1$	$4 - 2 \cdot (\cos(\frac{2\pi}{n}j_1) + \cos(\frac{2\pi}{n}j_2))$ $j_1, j_2 = 0, \dots, n-1$	$\frac{1-\cos(\frac{2\pi}{n})}{2}$

Tab. 4.1: Spectra of different graph classes. Numbers in brackets indicate the multiplicity of eigenvalues and are stated if an eigenvalue occurs more often than once.

which is much smaller than F_{de}.

The main disadvantage of the OPT scheme is the fact that we need to know all eigenvalues of the graph. Their calculation for arbitrary graphs is very time-consuming, but they are known for some classes of graphs, which play an important role as computer networks. Table 4.1 lists the spectrum of several graph classes, some of them are taken from [CDS95]. We denote with K_n the complete graph of cardinality n and with $K_{n,n}$ the complete bipartite graph of cardinality $2n$. P_n, C_n and S_n represent the path, the cycle and the star of cardinality n. Furthermore, the spectra of the hypercube ($Q(d)$), the 2-dimensional grid with n^2 vertices (G_n) and the 2-dimensional torus with n^2 vertices (T_n) are listed. The last column of Table 4.1 contains the condition number (λ_2/λ_n) of the Laplacians of the above mentioned graphs. As we have seen in Chapter 2, the condition number determines the number of iterations for the FOS. If the condition number increases, then the number of iterations decrease.

In the previous section we introduced ADI-FOS, ADI-SOS and ADI-Chebyshev, now we introduce ADI-OPT.

Theorem 16: *Let G' be a graph and let $G = G' \times G'$ be the second Cartesian power of it. We denote with m the number of distinct eigenvalues of the Laplacian of G'. Applying the OPT scheme alternating along each component of the Cartesian product, we obtain a load balancing scheme ADI-OPT, which only needs $m-1$ iterations to balance the load on G.*

Proof: Applying OPT on G we get $w^i = \left(I - \frac{1}{\lambda_{i+1}}L\right) w^{i-1}$, for any $i \in \{1, \dots, \frac{(m-1)m}{2}\}$, where $\tilde{\lambda}_{i+1}$ is a nonzero eigenvalue of $L = I \otimes L' + L' \otimes I$ (we assumed here that G has $\frac{(m-1)m}{2}$ different eigenvalues). By combining OPT with ADI we obtain

$$w^i = \left(I - \frac{1}{\tilde{\lambda}'_{i+1}} I \otimes L'\right)\left(I - \frac{1}{\tilde{\lambda}'_{i+1}} L' \otimes I\right) w^{i-1},$$

where $\tilde{\lambda}'_{i+1}$ are the nonzero eigenvalues of $I \otimes L'$ resp. of $L' \otimes I$. Then

$$\begin{aligned} e^{m-1} &= \prod_{i=2}^{m}\left(I - \frac{1}{\tilde{\lambda}'_i} I \otimes L'\right)\left(I - \frac{1}{\tilde{\lambda}'_i} L' \otimes I\right) e^0 \\ &= \prod_{i=2}^{m}\left(I - \frac{1}{\tilde{\lambda}'_i} I \otimes L'\right)\left(I - \frac{1}{\tilde{\lambda}'_i} L' \otimes I\right) \sum_{k,j \neq 1} z_{kj} = 0, \end{aligned}$$

where z_{kj} are the corresponding eigenvectors of $L = I \otimes L' + L' \otimes I$.

The Laplacian of $G' \times G'$ may have up to $\frac{m(m+1)}{2}$ distinct eigenvalues, i.e. the OPT scheme needs that many iterations, whereas the new ADI-OPT scheme uses only $m-1$ distinct eigenvalues of the Laplacian of G to eliminate all eigenvectors and to reduce the error to zero.

4.3 Experimental Results

In this section, we present experimental results concerning the ADI and OPT schemes. These results are based on experiments executed by Robert Preis. Since the behavior of ADI-SOS is similar to ADI-FOS, we discuss only the results concerning ADI-FOS. First, performing several experiments we try to find out, in which order the $m-1$ distinct non-zero eigenvalues $\tilde{\lambda}_2 < \tilde{\lambda}_3 < ... < \tilde{\lambda}_m$ in the OPT scheme should be used throughout the iterations. As stated before, the load after $m-1$ iterations is independent of the order chosen, but one may get trapped in numerical instable conditions with some orders.

We present the results of the OPT scheme with different choices of the orders in Figure 4.1. It shows the load balancing on a path of 32 nodes, for which the most difficult numerical problems can be observed. The path of 32 vertices has $m = 32$ distinct eigenvalues. Therefore, the balancing process is stopped after $m-1 = 31$ iterations. The initial load distribution is RANDOM (the shown results are for one random distribution; the experiments have been done with 100 random distributions and they all exhibited the same behavior). In RANDOM, $100 * |V|$ load elements are randomly distributed among the $|V|$ nodes. The eigenvalues are sorted in increasing order ($\tilde{\lambda}_2$, $\tilde{\lambda}_3$, $\tilde{\lambda}_4$, ...), decreasing order ($\tilde{\lambda}_m$, $\tilde{\lambda}_{m-1}$, $\tilde{\lambda}_{m-2}$, ...), sorted starting from the center ($\tilde{\lambda}_{\frac{m}{2}}$, $\tilde{\lambda}_{\frac{m}{2}-1}$, $\tilde{\lambda}_{\frac{m}{2}+1}$, $\tilde{\lambda}_{\frac{m}{2}-2}$, $\tilde{\lambda}_{\frac{m}{2}+2}$, ... for even m and $\tilde{\lambda}_{\frac{m-1}{2}}$, $\tilde{\lambda}_{\frac{m-1}{2}+1}$, $\tilde{\lambda}_{\frac{m-1}{2}-1}$, $\tilde{\lambda}_{\frac{m-1}{2}+2}$, $\tilde{\lambda}_{\frac{m-1}{2}-2}$, ... for odd m), or ordered by

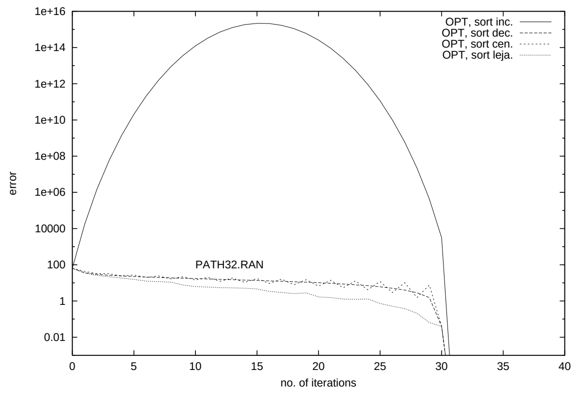


Fig. 4.1: The optimal scheme OPT on a path of 32 nodes with sorting of the eigenvalues in increasing, decreasing, center-started and by Leja Points order. The initial load distribution is random.

the Leja Points. Many people from the numerical area are dealing with orders for such problems and the last order was motivated by [Rei91].

The results show that with an increasing order of eigenvalues the error becomes very high after a few iterations and the final error is about 0.01, due to numerical problems with the high error numbers. The best behavior can be observed by using the Leja Points order. The eigenvalues $\tilde{\lambda}_2, \ldots, \tilde{\lambda}_m$ are sorted to a new sequence $\bar{\lambda}_2, \ldots, \bar{\lambda}_m$, such that $\bar{\lambda}_2 = \tilde{\lambda}_m$ and $\bar{\lambda}_i$, $i = 3, \ldots, m$, satisfies the property that $\prod_{j=2}^{i-1} \left| 1 - \frac{\bar{\lambda}_i}{\bar{\lambda}_j} \right| \bar{\lambda}_i$ is maximized. In this case, the final error is almost zero, although the error changes between increasing and decreasing for each iteration due to the changing order of high and low eigenvalues.

We compared the new optimal scheme OPT with the traditional First-Order scheme (FOS) and the more complicated optimal scheme OPS from [DFM99]. Therefore, experiments on a hypercube of dimension 6 (Figure 4.2) and on an 8x8 torus (Figure 4.3) have been done. In addition to the RANDOM initial distribution, there has also been used PEAK, in which one node has an initial load of $100 * |V|$ and the others have a load of 0. The balance process stopped after k iterations when the error $||w^k - \overline{w}||_2$ has been less than 10^{-6}. The hypercube of dimension 6 has $m = 7$ and the 8x8 torus has $m = 13$ distinct eigenvalues. The results show that both optimal schemes behave similar to FOS for the first iterations, but then they suddenly drop down to 0 after $m - 1$ iterations.

In the following, we will show results for the ADI scheme for the 16x16 torus, which is the Cartesian product of two cycles with 16 vertices each. We chose this example, because the torus often occurs as a network structure in parallel processing, but also other graphs like those in Table 4.1 can be used. First, we will show how the flow depends on the α chosen for the ADI-FOS scheme in Table 4.2, started on the PEAK initial load distribution (one node has a load of $100 * |V|$, the others a load of 0). The number of iterations as well as the l_1, l_2 and l_∞ norms of the flow are listed. Again, the balancing process stopped

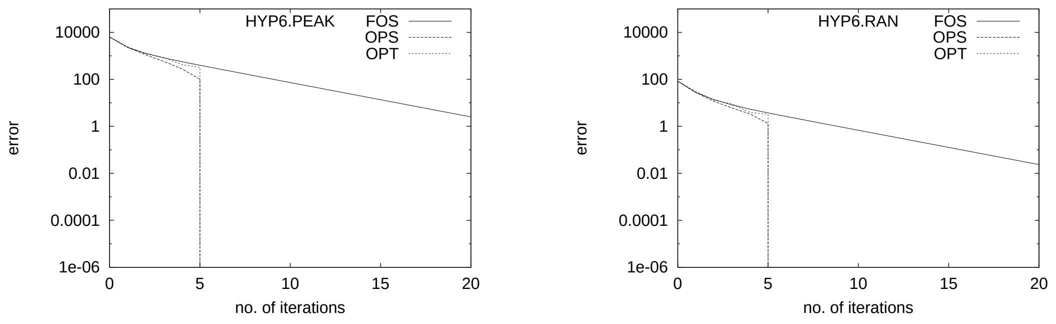


Fig. 4.2: Iterative Diffusion Load Balancing on a hypercube of dimension 6 with initial PEAK (left) and RANDOM (right) load distribution.

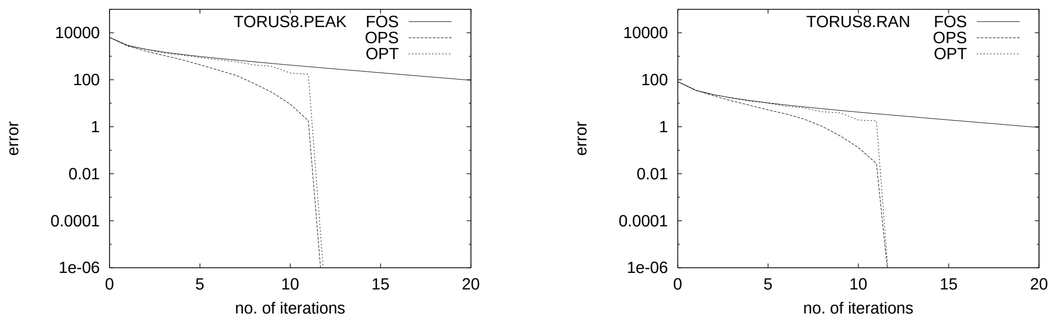


Fig. 4.3: Iterative Diffusion Load Balancing on an 8x8 torus with initial PEAK (left) and RANDOM (right) load distribution.

when error $||w^k - \overline{w}||$ was less than 10^{-6}. The results confirm the theoretical observations, i.e. with decreasing values of α the flow becomes the l_2-minimal flow, but also the number of iterations increases dramatically.

In the following we present the results of a number of experiments in order to compare the FOS and OPT schemes with the ADI and MDI modification schemes, started on the PEAK (Table 4.3) and RANDOM (Table 4.4; $100 * |V|$ load items are randomly placed on

	$\alpha =$					
norm/iter.	0.49	$opt = 0.4817$	0.4	0.2	0.1	0.01
l_1	627200	355082.51	207242.41	204800	204800	204800
l_2	52500.69	39311.89	21361.38	18188.09	17967.10	17919
l_∞	19066.10	16743.38	10828.53	7637.55	6908.24	6422.10
iter.	528	291	349	708	1427	14366

Tab. 4.2: The flow of ADI-FOS with different values for α on a 16x16 torus.

norm/iter.	FOS	OPT	ADI-FOS	ADI-OPT	MDI-FOS	MDI-OPT
l_1	204800	204800	355082.51	204800	205663.31	2047800
l_2	17918.62	17918.62	39311.89	20235.24	23699.66	19993.43
l_∞	6375	6375	16743.32	9926.57	12185.46	9750.56
iter.	578	40	291	8	291	8

Tab. 4.3: The flow of ADI and MDI for initial PEAK load distribution on a 16x16 torus.

norm/iter.	FOS	OPT	ADI-FOS	ADI-OPT	MDI-FOS	MDI-OPT
l_1	2580.8	2580.8	4122.63	2726.69	3047.82	2715.94
l_2	141.90	141.90	228.14	152.22	170.88	151.91
l_∞	19.40	19.40	29.93	19.72	22.26	19.66
iter.	458	40	229	8	229	8

Tab. 4.4: The flow of ADI and MDI for initial RANDOM load distribution on a 16x16 torus.

the nodes) initial load distribution. We only show one example of RANDOM distribution, because all RANDOM distributions exhibited the same behavior. As stated in Chapter 2, the FOS and OPT schemes both compute the same l_2-minimal flow, whereas the flow in the ADI and MDI case may differ from each other. As we have proved before, neither of their flows is minimal in the l_2 norm and they are also worse in respect to the l_1 and l_∞ norms of our test cases. Furthermore, the flow of the ADI-OPT scheme is much smaller than the flow of the ADI-FOS scheme in all norms and it is only a small fraction larger than the l_2-minimal flow. The flow of the MDI scheme improves over the flow of the ADI scheme in all test cases. Therefore, it should be preferably used for practical applications.

The number of iterations for the ADI and MDI schemes are the same and are always smaller than the examples without ADI or MDI. As we have proven before, the upper bound on the number of iterations for FOS is twice as high as for ADI-FOS. In the experiments ADI-FOS, compared to FOS, reduce to the half the number of iterations for the RANDOM case and almost reduce them to the half for the PEAK case. For the optimal scheme, the number of iterations depend on the number of distinct eigenvalues as shown in the previous section, which is 41 for the 16x16 torus and 9 for the cycle of 16 vertices (the basis graph which product results in the 16x16 torus). Therefore, the number of iterations reduces from 40 to 8 if the ADI or MDI schemes are used.

4.4 Sparse Network Topologies with Small Spectrum

As described in Section 4.2, by applying the optimal scheme OPT on a graph, we only need $m-1$ iterations to balance the load, where m is the number of different eigenvalues of the Laplacian of the graph. Additionally, in any iteration, a node has to communicate with all of its neighbors, so the total cost of the load balancing algorithm depends on the distinct eigenvalues of the graph and on its vertex degree. The number of steps is, in fact, the product of both.

Sparse networks having a small number of different eigenvalues can be defined as virtual topologies in a bus system. In such a system, every processor can communicate with any other processor, but in order to avoid high communication costs, we allow each processor to communicate only with a restricted number of other processors, i.e. we are interested in a sparse virtual topology. In addition, the virtual network should be well-suited for load balancing algorithms and therefore, a sparse graph having a small spectrum satisfies the requirements.

The set of the eigenvalues of a graph is an important algebraic invariant in other scientific fields also. See [Big93, Chu97] or [CDS95] for a selection of results in this area. The size of the spectrum of a graph is also correlated to its symmetry properties: the only graph having two distinct eigenvalues is the complete graph and its automorphism set is as rich as possible - the symmetric group. Graphs having three distinct eigenvalues are called strongly regular; their diameter is 2 and they posses many interesting properties [Big93, Hub75].

In this section, we present scalable and non-scalable network topologies having a small vertex degree and only $O(\log^k(n))$ different eigenvalues, where n is the cardinality of the considered graphs and $k \in \mathbb{N}$ is a small constant.

4.4.1 Non-Scalable Topologies with a Small Laplacian Spectrum

In the past, several papers have been published about well-structured graphs. Consider for example the hypercube of dimension d. It has 2^d vertices, $d \cdot 2^{d-1}$ edges, and a diameter resp. vertex degree of d (see Figure 4.4). The hypercube has $d+1$ distinct eigenvalues (already described in Section 4.2) and a large application as an interconnection topology. Other graphs, such as cliques, complete bipartite graphs or the star, have only 2 resp. 3 distinct eigenvalues, but due to their high density (e.g. cliques and complete bipartite graphs) or to bottlenecks in the graph (e.g. the star) they are ill-suited as interconnection topologies.

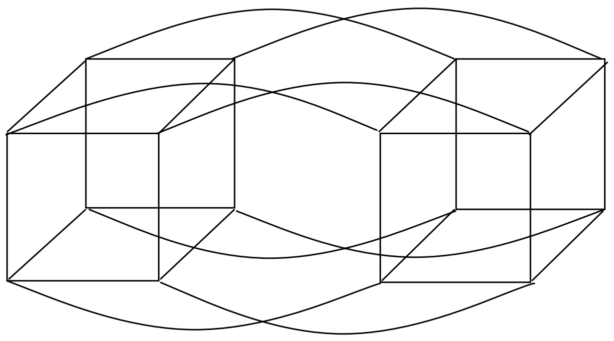


Fig. 4.4: The hypercube of dimension 4.

There exist some other graphs with even a better relation between number of vertices, vertex degree, diameter and number of different eigenvalues than the hypercube. One of them is the Petersen graph, which has 10 vertices, a vertex degree of 3, diameter 2 and 3 different eigenvalues. Another one is the $Cage(6, 6)$, which has 62 vertices, vertex degree 6, a diameter of 3 and only 4 different eigenvalues. Both graphs are presented in Figure 4.5.

A family of graphs with a very good behavior is the family of star graphs. The star graph $S(d)$ of dimension d has $d!$ vertices and is defined as follows. A vertex of $S(d)$ is denoted by the sequence of length d, $(i_1, i_2, \ldots, i_d)$ with $i_j \in \{1, \ldots, d\}$ and $i_j \neq i_l$ for any $j, l \in \{1, \ldots, d\}$, $j \neq l$. Two nodes $(i_1, \ldots, i_d)$ and $(i'_1, \ldots, i'_d)$ are connected with each other iff there exists a $j \in \{1, \ldots, d\}$ such that $i_1 = i'_j$ and $i_l = i'_l$ for any $l \in \{2 \ldots, j-1, j+1, \ldots, d\}$. Due to this definition, $S(2)$ has 2 vertices connected by an edge, $S(3)$ is a cycle with 6 vertices and $S(4)$ is represented in figure 4.6.

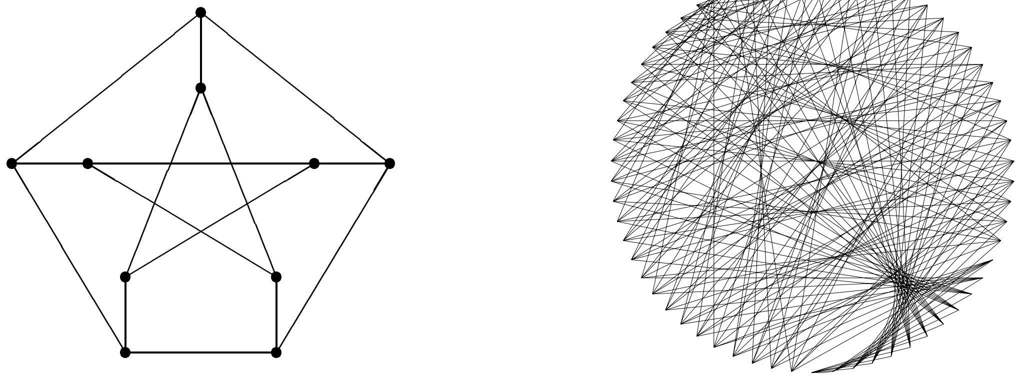


Fig. 4.5: The Petersen graph (left) and the $Cage(6, 6)$ (right)

Now, let us analyze the vertex degree, diameter and the number of different eigenvalues

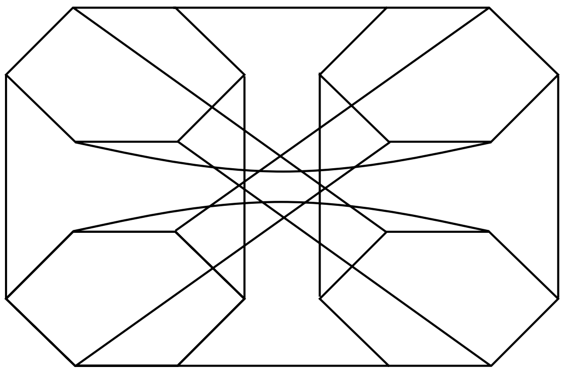


Fig. 4.6: The star graph of dimension 4.

of the star graph. It follows directly from the definition, that $S(d)$ is regular of degree $d-1$. The diameter of it is $\lfloor\frac{3(d-1)}{2}\rfloor$ as calculated in [AHK87]. Using a result of Flatto, Odlyzko and Wales the following theorem can be stated [FOW85].

Theorem 17: *The eigenvalues of the Laplacian of the star graph $S(d)$ are the integers $0, 1, 2, \ldots, 2(d-1)$.*

From Theorem 17 immediately follows, that the star graph of dimension d has only $2d-1$ different eigenvalues. Note, that the well-known lower bound on the size of the spectrum of a graph G of cardinality n and maximum vertex degree deg is $\Omega(\frac{\log(n)}{\log(deg)})$. The star graph is one of the few graphs, which achieve this lower bound w.r.t. the number of different eigenvalues. Since the star graph has also a small vertex degree, they are best suited for load balancing applications. Unfortunately, these graphs are not scalable. A graph class is called scalable if for any $n \in I\!N$, there exists a graph of cardinality n belonging to this graph class.

An other well-known interconnection topology is the Butterfly graph, which is designed to have many favorable properties for distributed computing, such as small vertex degree, small diameter and large connectivity. The d dimensional Butterfly $BF(d)$ has $(d+1)2^d$ vertices and $d2^{d+1}$ edges. The vertices correspond to pairs (q, i) where i is the level of the node, $0 \leq i \leq d$ and q is a d-bit binary number that denotes the row of the node. Two vertices (q, i) and (q', i') are connected by an edge iff $i' = i + 1$ and either q and q' are identical, or q and q' differ in precisely the i'^{th} bit. If q and q' are identical, then the edge is said to be a *straight edge*. Otherwise, the edge is a *cross edge*. As an example, we represented the Butterfly of dimension 3 in Figure 4.7. It follows immediately from this

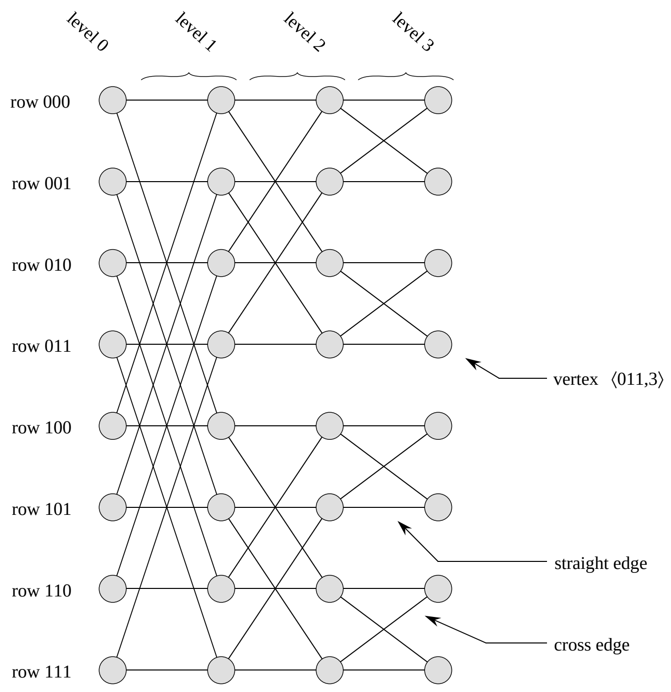


Fig. 4.7: The Butterfly graph of dimension 3.

definition that $BF(d)$ has a maximum vertex degree of 4 for any $d \in I\!N$. In [EKM01] and [Sch01] the eigenvalues are computed.

Theorem 18: *The Laplacian spectrum of the Butterfly $BF(d)$ has the form*

$$S_{BF(d)} = \left\{4 - 4\cos\left(\frac{\pi i}{j+1}\right) \mid 0 \le i \le j \le d\right\} \cup \left\{4 - 4\cos\left(\frac{\pi(2i-1)}{2j+1}\right) \mid 1 \le i \le j \le d\right\}$$

Hence, the number of different eigenvalues of this graph is $O(\log^2(n))$, where n is the cardinality of $BF(d)$. This graph is therefore well-suited for load balancing applications.

A network related to the Butterfly is the so called wrapped Butterfly of dimension d, where the first and the last levels of $BF(d)$ are merged into a single level. In particular, vertex $(q, 0)$ is merged into vertex (q, d) for each $q \in \{0, \ldots, 2^d - 1\}$. The result is a d-level graph with $d2^d$ nodes, each of degree 4. Two vertices (q, i) and (q', i') are connected by an edge iff $i' \equiv i \mod d$ and either $q = q'$ or q and q' differ in the i'^{th} bit. Due to the result of Schmidt in [Sch01] the following theorem can be stated.

Theorem 19: *The Laplacian spectrum of the wrapped Butterfly of dimension d, $wBF(d)$ has the form*

$$S_{wBF(d)} = \left\{4 - 4\cos\left(\frac{\pi i}{j+1}\right) \mid 1 \le i \le j \le d\right\} \cup \left\{4 - 4\cos\left(\frac{2\pi i}{d}\right) \mid 0 \le i \le d-1\right\}$$

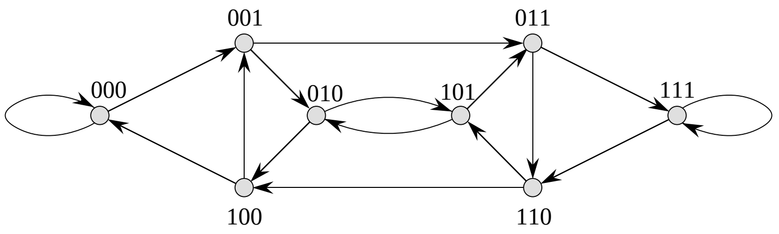


Fig. 4.8: The directed de Bruijn graph of dimension 3.

Similar to the Butterfly, the wrapped Butterfly of dimension d has also $O(\log^2(n))$ different eigenvalues, where n is the cardinality of the graph.

The Butterfly and the wrapped Butterfly possess structures that are very similar to that of the hypercube and hence, it should not be surprising that they can efficiently simulate many hypercube computations. Now we describe an other network that appear to have little in common with the hypercube, but is at least as good as the Butterfly at simulating hypercube computations. This network is the de Bruijn graph.

The directed d-dimensional de Bruijn graph consists of 2^d vertices and 2^{d+1} directed edges. Each node corresponds to a d-bit binary string, and there is a directed edge from each node $(i_1, i_2, \dots, i_d)$ to $(i_2, i_3, \dots, i_d, i_1)$ and to $(i_2, i_3, \dots, i_d, \overline{i_1})$. The edges of the first kind are called *shuffle edges*, while the edges of the second kind are *shuffle-exchange edges* (see also Figure 4.8 for an example). By replacing each directed edge by an undirected edge we obtain the (undirected) de Bruijn graph, which is regular of degree 4. Note, that this definition allows 2 loops at the vertices $(0, 0, \dots, 0)$ and $(1, 1, \dots, 1)$ and one double edge between the vertices $(0, 1, 0, 1, \dots)$ and $(1, 0, 1, 0, \dots)$. In [DT98], Delorme and Tillich computed the eigenvalues of the de Bruijn graph and stated the following theorem.

Theorem 20: *The Laplacian eigenvalues of the de Bruijn graph of dimension d are of the form*

$$4 - 4\cos\left(\frac{\pi i}{d+1}\right),$$

where $i \in \{1, \dots, d\}$.

Hence, the de Bruijn graph of dimension d, $d \in I\!N$ has also $O(\log^2(n))$ different eigenvalues, with n being the cardinality of the graph.

4.4.2 Scalable Topologies with a Small Laplacian Spectrum

The graphs presented above have the main disadvantage that they are not scalable. In this subsection, we describe a method how to construct families of scalable sparse graphs

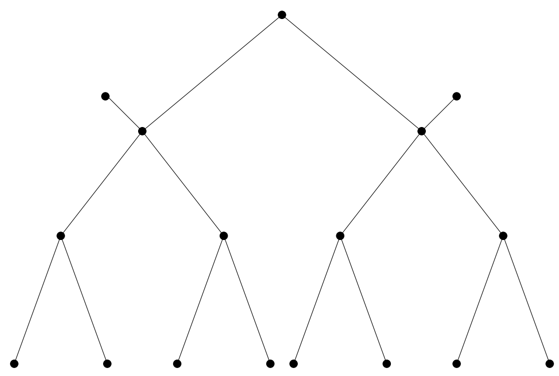


Fig. 4.9: The graph $T_{(2,17)}$.

having a small number of different eigenvalues. As a first example define the graph class $T_{(d,n)}$ as follows.

Definition 7: *The tree $T_{(d,n)}$ is a rooted tree defined recursively. $T_{(d,1)}$ contains only the root vertex. For $n < d+1$, $T_{(d,n)}$ is a star with one root and $n-1$ leaves connected to it. For $n > d$, $T_{(d,n)} = (V, E)$ is constructed as follows. Let $s = \lfloor \frac{n-1}{d} \rfloor$ and let $q = n - 1 - ds$. Let $(V_1, E_1), ..., (V_d, E_d)$ be d disjoint copies of $T_{(d,s)}$ with roots $r_1,...,r_d$. Then $V = \{r\} \cup \{v_1, ..., v_q\} \cup \bigcup_{i=1}^{d} V_i$ and $E = \{(r, v_i) \mid i = 1...q\} \cup \{(r, r_i) \mid i = 1..d\} \cup \bigcup_{i=1}^{d} E_i$.*

Informally, constructing a tree $T_{(d,n)}$ involves setting one vertex r as a root, then dividing the remaining $n-1$ vertices evenly and constructing a number of copies of $T_{(d,s)}$. The roots of these copies are connected to r. The remaining vertices are connected as vertices with degree 1 to r (see also Figure 4.9). We can observe, that the graph $T_{(d,n)}$ has a maximum vertex degree of at most $2d+1$. In [EKM01] the following theorem has been shown.

Theorem 21: *The Laplacian of the graph $T_{(d,n)}$ has at most $O\left(\left(\frac{\log n}{\log d}\right)^2\right)$ different eigenvalues.*

The product of the maximum vertex degree and the number of different eigenvalues is $O\left(d\left(\frac{\log n}{\log d}\right)^2\right)$ and hence, the optimal diffusion scheme OPT would work very fast on this topology. However, the graph $T_{(d,n)}$ has a tree-like structure, and trees are extremely ill-suited for load balancing applications because they contain a bottleneck at the root and have poor connectivity properties. To overcome this weak point, we give another construction of a graph $H_{(n)}$ with $O\left((\log n)^3\right)$ distinct eigenvalues, which is much better suited as a topology for load balancing.

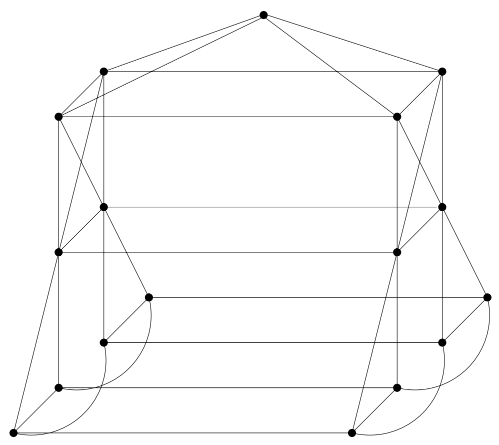


Fig. 4.10: The graph $H_{(17)}$.

Definition 8: *Every graph $H_{(n)}$ has a set of distinguished vertices (core) denoted by $C(H_{(n)})$. The graph $H_{(n)}$ is defined recursively as follows. $H_{(1)} = C(H_{(1)}) = K_1$, $H_{(2)} = C(H_{(2)}) = K_2$. To obtain $H_{(n)} = (V, E)$, $n > 2$, first construct two copies of $H_{(\lfloor \frac{n-1}{2} \rfloor)}$ and connect the corresponding vertices between these copies. The remaining* 1 *or* 2 *vertices of $H_{(n)}$ form its core connected together and they are interconnected with all vertices from the cores of both $H_{(\lfloor \frac{n-1}{2} \rfloor)}$'s.*

For an example of $H_{(17)}$, see Figure 4.10. Due to definition 8, the maximum vertex degree of $H_{(n)}$ is at most $\log(n) + 5$. In [EKM01], the following theorem is stated.

Theorem 22: *The graph $H_{(n)}$ has at most $O(\log^3 n)$ distinct eigenvalues.*

Although, $H_{(n)}$ has more different eigenvalues than $T_{(d,n)}$, $H_{(n)}$ has a much larger connectivity and therefore, it is better suited as an interconnection topology.

4.4.3 Scalable Topologies with a Small Adjacency Spectrum

The number of different eigenvalues of the graphs presented in the previous subsection is polylogarithmic w.r.t. the cardinality of the graphs, but they do not achieve the lower bound described at the beginning of this section. Now we present some other scalable families of graphs and show that the size of the adjacency spectrum of some of these graphs achieve the lower bound previously mentioned. However, we were not able to achieve the lower bound for the size of the Laplacian spectrum of scalable graphs.

Let $G_{(n)}$ be defined as follows (see also figure 4.11 for an example).

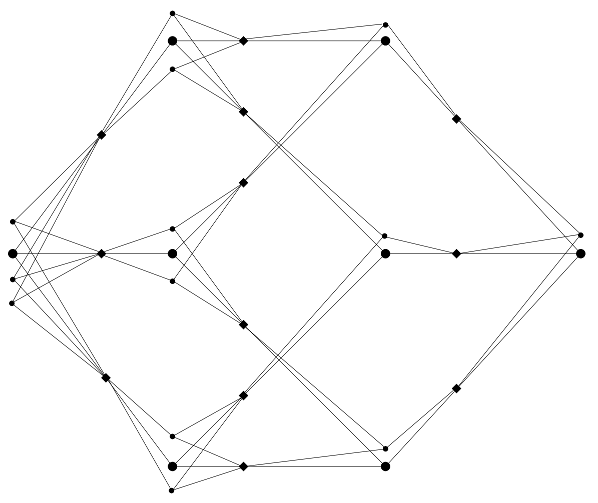


Fig. 4.11: The graph $G_{(33)}$.

Definition 9: *Let d ($d > 0$) satisfy the inequality $2^{d-1}(d+2) \leq n < 2^d(d+3)$. Let $\{\Delta_i\}_{i=0}^{d}$ be a sequence defined as follows:*

$$\Delta_i = \begin{cases} 0 & \text{for } i > \left\lfloor \frac{d}{2} \right\rfloor \\ \big(n - 2^{d-1}(d+2)\big) \mod 2^d & \text{for } i = \left\lfloor \frac{d}{2} \right\rfloor \\ \Delta_{i+1} \mod \binom{d}{i+1} & \text{for } 0 \leq i < \left\lfloor \frac{d}{2} \right\rfloor \end{cases}$$

where $a \mod b = a - b \cdot \left\lfloor \frac{a}{b} \right\rfloor$.

Let $Q(d)$ be the d dimensional hypercube. We shall refer to the vertices as binary strings from $\{0,1\}^d$ and define the k^{th} level of $Q(d)$ as $\mathcal{L}_k = \{x \in \{0,1\}^d \mid \#_1(x) = k\}$.

The graph $G_{(n)}$ is defined as follows. Consider the graph $S(Q(d))$ with $2^{d-1}(d+2)$ vertices obtained from the hypercube $Q(d)$ by subdivision of each edge. For each node x from the original graph $Q(d)$ add a set $\mathcal{V}_x$ of isolated vertices of cardinality $|\mathcal{V}_x| = \left\lfloor \frac{\Delta_k}{\binom{d}{k}} \right\rfloor + \left\lfloor \frac{n-2^{d-1}(d+2)}{2^d} \right\rfloor$ where $x \in \mathcal{L}_k$. For each $y \in \mathcal{V}_x$ add edges (y,v) for all edges (x,v) from $S(Q(d))$.

Since the hypercube of cardinality n has a diameter of $\log(n)$, the graph $G_{(n)}$ has a diameter of at most $2\log(n)$. The definition of this graph also implies that it has a maximum vertex degree of at most $3\log(n) + o(\log(n))$. In [EKM01], the number of different eigenvalues of this graph has also been determined.

Theorem 23: *The adjacency matrix of $G_{(n)}$ has $O(\log^2(n))$ different eigenvalues.*

Unfortunately, a similar theorem could not be obtained for the Laplacian of these graphs and therefore, we are not able to determine how fast the optimal diffusion load balancing scheme OPT converges on these topologies. Note that these graphs are better suited as interconnection topologies than the graph classes defined in the previous subsection. However, the lower bound presented at the beginning of this section is still not tight for $G_{(n)}$. To achieve this, we can use a simple trick in order to reduce the number of different eigenvalues of the adjacency matrix of $G_{(n)}$ to $O(\log(n))$.

If we assign the value $\sqrt{|\mathcal{V}_x|}$ to each edge (y, v) of $G_{(n)}$, where $y \in \mathcal{V}_x$ and x is a vertex of the original hypercube $Q(d)$, then the lower bound w.r.t. the size of the spectrum will be nearly tight. The resulting graph is a weighted graph, however, the polynomial-based diffusion schemes also work for weighted graphs.

Similar transformations can also be applied for other graphs. Use, for example, the star graph instead of the hypercube to perform the described transformations and then, we can immediately construct a scalable graph class, for which the lower bound w.r.t. the size of its adjacency spectrum is achieved.

4.5 Summary

In this chapter, we used structural and spectral properties of graphs to develop new load balancing algorithms. In Section 4.1, we proposed a mixture of diffusion and dimension exchange for Cartesian product of graphs. We showed that this new algorithm works faster than classical diffusion and, for products of two isomorphic graphs, we can achieve an improvement factor of 2. In Section 4.2, we presented a new optimal scheme having a very simple construction. In Section 4.3, we confirmed our theoretical results by several experiments. In the last section, we described the construction of scalable and non-scalable sparse network topologies, which are well-suited for load balancing applications.

5. OPTIMAL DIFFUSION MATRICES

In the previous chapter, we presented new load balancing algorithms for homogeneous networks. In this chapter, we consider inhomogeneous schemes described by edge-weighted graphs. The goal is to find edge-weights such that the condition number of the resulting Laplacian is maximized among all Laplacians having the same communication structure.

First, we concentrate on edge-transitive graphs and show that, for these graphs, the condition number is maximized if all edges have the same weight. Second, we prove for Cayley graphs that edges generated by the same generator must be of equal weight in order to maximize the condition number of their Laplacians. For Cartesian products of graphs, we compute edge-weights that improve (but not necessarily maximize) the condition number of the corresponding Laplacians. Finally, we compute optimal weights for the edges of Cube Connected Cycles and other hypercubic networks. To confirm our theoretical results, in Section 5.3, we present experimental results concerning different edge-weight scenarios of the mentioned graph types and show some dependencies between edge-weights and convergence rate.

5.1 Basic Definitions

Let $G = (V, E)$ be a connected, weighted, undirected graph with $|V| = n$ nodes and $|E| = N$ edges. Let $c_{i,j} \in I\!R^N$ be the *weight* of edge $\{v_i, v_j\} \in E$, $w_i \in I\!R$ be the load of node $v_i \in V$ and $w \in I\!R^n$ be the vector of load values. $\overline{w} := \frac{1}{n}(\sum_{i=1}^{n} w_i)(1, \ldots, 1)$ denotes the vector of the corresponding average load.

The weighted adjacency matrix of G is defined by $B \in I\!R^{n\times n}$, where column/row i of B contains $c_{i,j}$ for any v_j and v_i neighbors in G. As already mentioned in Chapter 2, the weighted Laplacian is defined similar to the unweighted case, $L := D - B$, where $D \in I\!N^{n\times n}$ contains the weighted degrees as diagonal entries, e. g. $D_{i,i} = \sum_{\{v_i,v_j\}\in E} c_{i,j}$, and 0 elsewhere.

We consider the following local iterative load balancing algorithm that generalizes equation (2.2) and requires communication with adjacent nodes only

$$\begin{aligned}\forall\, e = \{v_i, v_j\} \in E :\ y_{i,j}^{k-1} &= \alpha c_{i,j}(w_i^{k-1} - w_j^{k-1}) \\ x_e^k &= x_e^{k-1} + \delta_{i,j} y_{i,j}^{k-1} \\ w_i^k &= w_i^{k-1} - \sum_{e=\{v_i,v_j\}\in E} y_{i,j}^{k-1}.\end{aligned} \tag{5.1}$$

Again, $\delta_{i,j}$ represents the arbitrarily assigned edge direction, $\delta_{i,j} y_{i,j}^k$ describes the amount of load sent via edge $e = \{v_i, v_j\}$ in step k, x_e^k is the load sent via edge e added up until iteration k and w_i^k is the load of the node v_i after the k^{th} iteration. If a directed edge is pointing from v_i to v_j (e.g. $i > j$), then $\delta_{i,j} = 1$ otherwise $\delta_{i,j} = -1$. Note, that $\delta_{i,j} = -\delta_{j,i}$ and therefore $\delta_{i,j} y_{i,j}^k = \delta_{j,i} y_{j,i}^k$ for any pair of $\{v_i, v_j\} \in E$. Computing the flow x_e^k can be skipped in case of a one-step model since there the load is immediately moved and no monitoring needs to be done. Throughout this thesis, however, we assumed that we always use the three-step model, in which first a balancing flow is calculated, second a scheduling is computed, and third the load is moved accordingly.

By generalizing the diffusion matrix $M = I - \alpha L$ to edge-weighted graphs, equation (5.1) can be written in matrix notation as $w^k = Mw^{k-1}$. Here, M contains $\alpha c_{i,j}$ at position (i, j) for every edge $e = \{v_i, v_j\}$, $1 - \sum_{e=\{v_i,v_j\}\in E} \alpha c_{i,j}$ at diagonal entry i, and 0 elsewhere. Similar to the unweighted case, α has to be chosen such that $1 = \mu_1 \geq \mu_2 \geq \ldots \geq \mu_n > -1$, where we denoted with μ_i, $i \in \{1, \ldots, n\}$, the eigenvalues of M. It can be also shown that the fastest convergence is achieved by choosing $\alpha = \frac{2}{\lambda_2+\lambda_n}$. Note that throughout this section $\lambda_1, \ldots, \lambda_n$ denote the eigenvalues of the weighted Laplacian L. Again, $\gamma = \max\{|\mu_2|, |\mu_n|\} < 1$ is the second largest eigenvalue of M according to absolute values and call it the *diffusion norm* of M.

We can define the *second order scheme* in the edge-weighted case also by

$$w^1 = Mw^0 \text{ and } w^k = \beta M w^{k-1} + (1-\beta) w^{k-2},\ k = 2, 3, \ldots$$

with β being a fixed parameter, whereby fastest convergence is archived for $\beta = \frac{2}{1+\sqrt{1-\mu_2^2}}$. The Chebyshev method [DFM99] can also be generalized in a similar manner. According to the unweighted case, we can generalize Lemma 1, 2, and 3 for weighted graphs also. The result of Lemma 3 shows that FOS and SOS converge faster if the condition number is higher. Therefore, by using edge-weighted graphs it is possible to increase the condition number of the Laplacian and to reduce the number of steps needed to compute a balancing flow, which distributes the load in the network.

Similar to the unweighted case, an l_2 optimal flow is represented by the minimal flow with respect to the weighted Euclidian norm, i. e. the solution to the problem

$$\text{minimize } \|x^k\|_2 = \sqrt{\sum_{\{v_i,v_j\}\in E} \frac{(x_{\{v_i,v_j\}}^k)^2}{c_{i,j}}} \text{ over all balancing flows } x^k.$$

Here, $c_{i,j}$ represent the weight assigned to the edge $\{v_i, v_j\} \in E$. Then, we can state the following Lemma [DFM99].

Lemma 13: *On any (weighted) graph G, FOS and the SOS compute an l_2-minimal balancing flow.*

In [DFM99], Lemma 13 has been proved for polynomial-based load balancing schemes in a general form.

5.2 Graph Classes and Interconnection Topologies

In this section we deal with general graph classes like edge-transitive graphs, Cayley graphs and Cartesian products of graphs as well as with interconnection topologies like grid (G), torus (T), Cube Connected Cycles (CCC), Butterfly (BF) and de Bruijn (DB) networks. These interconnection topologies are designed to have many favorable properties for distributed computing, e.g. small vertex degree and diameter and large connectivity. In the present thesis, we focus our attention on computing optimal edge-weights for these graphs in order to maximize the condition number of the corresponding Laplacian. First, let us concentrate on some simple graphs like cycles, hypercubes, complete graphs or the star. All these graphs are edge-transitive. In other words, for any pair of edges $\{u, v\}$ and $\{u', v'\}$ there is an automorphism σ such that $\sigma(u) = u'$ and $\sigma(v) = v'$. An *automorphism* of a graph is a one-to-one mapping of nodes onto nodes such that edges are mapped onto edges. To show that for all edge-transitive graphs the maximal condition number is achieved if all edges have the same weight, the following lemma is useful.

Lemma 14: *Let $L_0, L_1, \ldots, L_m \in I\!R^{n\times n}$ be Laplacian matrices of weighted graphs $G_0, \ldots, G_m$, all with the same adjacency structure, and let $L_0 = \frac{L_1+\cdots+L_m}{m}$. If we denote with $\lambda_2(L_i)$ and $\lambda_n(L_i)$ the second smallest and largest eigenvalues of the Laplacians L_i, $i \in \{0, \ldots, m\}$, then $\lambda_2(L_o) \geq \min\{\lambda_2(L_1), \ldots, \lambda_2(L_m)\}$ and $\lambda_n(L_0) \leq \max\{\lambda_n(L_1), \ldots, \lambda_n(L_m)\}$.*

Proof: We know that $(1, \ldots, 1)^t$ is an eigenvector of L_i, $0 \leq i \leq m$, having the eigenvalue 0. For all graphs G_i we denote the number of vertices with n and the number of edges with N. Furthermore, we denote with $c^l_{i,j}$ the weights of the edges $\{v_i, v_j\}$ of G_l. We assume w.l.o.g. that $\lambda_2(L_1) \leq \lambda_2(L_2) \leq \cdots \leq \lambda_2(L_m)$. Let $(z_1, \ldots, z_n)^t$ be an eigenvector of L_0 corresponding to the eigenvalue $\lambda_2(L_0)$. Then, using the Rayleigh coefficient we obtain

$$\begin{aligned}
\lambda_2(L_0) &= \frac{\sum_{\{v_i,v_j\}\in E_{G_l}}\left(\frac{\sum_{l=1}^{m} c^l_{i,j}}{m}(z_i-z_j)^2\right)}{\sum_{v_i\in V_{G_l}} z_i^2} = \frac{1}{m}\sum_{l=1}^{m}\frac{\sum_{\{u,v\}\in E_{G_l}} c^l_{i,j}(z_i-z_j)^2}{\sum_{v_i\in V_{G_l}} z_i^2}\\
&\geq \frac{1}{m}\sum_{l=1}^{m}\left(\min_{y\perp\mathbf{1}}\frac{\sum_{\{v_i,v_j\}\in E_{G_l}} c^l_{i,j}(y_i-y_j)^2}{\sum_{v_i\in V_{G_l}} y_i^2}\right) = \frac{1}{m}(\lambda_2(L_1)+\cdots+\lambda_2(L_m))\\
&\geq \lambda_2(L_1)
\end{aligned}$$

where y and $\mathbf{1} = (1,\ldots,1)$ are vectors of size n.

The second statement of the lemma can be obtained by replacing "$\leq$" with "$\geq$" and "min" with "max". $\square$

Now, let L_1 be the Laplacian of a weighted edge symmetric graph. Applying the lemma to the family of all matrices that can be obtained from L_1 by permuting rows and columns according to some automorphism of G, we obtain a Laplacian L_0 having the same non-diagonal entry for any edge. Due to Lemma 14, the condition number of L_0 will not be smaller than the condition number of L_1 and we can state the following theorem.

Theorem 24: *Let G be an unweighted, edge-transitive graph. Among all weighted graphs with G's adjacency structure, the condition number of the Laplacian is maximized for the one that has all edge-weights set to* 1.

In this thesis, we consider several graphs that can be viewed as Cayley graphs.

Definition 10: *Let G be any abstract finite group with identity* 1, *and let Ω be a set of generators for G with the properties $x \in \Omega \Rightarrow x^{-1} \in \Omega$ and $1 \notin \Omega$. The Cayley graph $\Gamma = \Gamma(G,\Omega)$ is a simple graph with vertex set $V_\Gamma = G$ and edge set $E_\Gamma = \{\{g,h\}|g^{-1}h \in \Omega\}$.*

An edge {h,k} is generated by a generator $\omega \in \Omega$, iff $h^{-1}k = \omega$ or $k^{-1}h = \omega$. We now show that edges of the same generator of Ω must have the same weight in order to achieve a minimal amount of iteration steps in diffusion algorithms.

Theorem 25: *Let Γ be a Cayley graph and let Ω be the set of its generators. The condition number of the Laplacian is maximized, if for any two edges $e = \{g,h\}$ and $e' = \{g',h'\}$ generated by the same generator $\omega \in \Omega$ the edge weights are equal.*

Proof: For each $g \in G$ we may define a permutation $\overline{g}$ of V_Γ by the rule $\overline{g}(h) = gh$, $(h \in G)$. This is an automorphism of Γ [Big93]. If there exists an edge between h and k generated by ω, then there also exists an edge between gh and gk generated by ω.

Assume $\omega^{-1} \neq \omega$ and let p be the smallest integer with the property $\omega^p = 1$. Then, ω generates cycles of length p where each vertex has an incident edge generated by ω and an other incident edge generated by ω^{-1}. Therefore, the number of edges generated by ω equals the number of vertices of Γ. Next, we have to show that for different g and g' the edge $\{h, k\}$ is mapped to different edges. Assume that $\{gh, gk\} = \{g'h, g'k\}$. Then $gk = g'k$ and $gh = g'h$ or $gk = g'h$ and $gh = g'k$. In the first case, we have a contradiction to the assumption that $g \neq g'$, while in the second case there is a contradiction to $\omega^{-1} \neq \omega$. Hence, we can use $|G|$ permutations to map each edge to every other edge and the theorem follows by lemma 14. If $\omega^{-1} = \omega$, using $|G|$ permutations causes each edge being mapped twice to every other edge in the graph and the theorem also follows by lemma 14. □

As a consequence of this theorem, edges belonging to the same dimension of a torus must have the same weight. On the other hand, a torus can be viewed as a Cartesian product of cycles. For a Cartesian product of two graphs G and H, however, we can state the following theorem.

Theorem 26: *Let G and H be two unweighted graphs and denote with $G \times H$ their Cartesian product. Let $\lambda_2(G)$ and $\lambda_2(H)$ be the second smallest eigenvalue of the Laplacian of G and H, respectively. W.l.o.g, assume $\lambda_2(G) \leq \lambda_2(H)$. Then, the diffusion schemes on $G \times H$ can be improved by assigning the weight $\frac{\lambda_2(H)}{\lambda_2(G)}$ to the edges of G and 1 to the edges of H.*

Proof: The second smallest eigenvalue of $G \times H$ is $\min\{\lambda_2(G), \lambda_2(H)\}$ and the largest eigenvalue of $G \times H$ has the form $\lambda_n(G) + \lambda_n(H)$. Let a be the weight of the edges of G. We have to maximize the function $\rho = \min\{\frac{a\lambda_2(G)}{a\lambda_n(G)+\lambda_n(H)}, \frac{\lambda_2(H)}{a\lambda_n(G)+\lambda_n(H)}\}$. The function $\frac{a\lambda_2(G)}{a\lambda_n(G)+\lambda_n(H)}$ is increasing, while $\frac{\lambda_2(H)}{a\lambda_n(G)+\lambda_n(H)}$ is decreasing in a. It follows that $a = \frac{\lambda_2(H)}{\lambda_2(G)}$ holds for a maximized ρ. □

Note that if both graphs G and H defined in Theorem 26 are edge-transitive graphs, then assigning weight $\frac{\lambda_2(H)}{\lambda_2(G)}$ to the edges of G and 1 to the edges of H maximizes the condition number of the Laplacian matrix.

Since the eigenvalues of a cycle of length n are $2 - 2\cos(\frac{2\pi j}{n})$, $0 \leq j < n$, we can state the following corollary.

Corollary 8: *Let T be the d-dimensional torus generated from the Cartesian product of d cycles of length $n_1 \leq n_2 \leq \cdots \leq n_d$. The polynomial-based diffusion algorithms have their fastest convergence rate if the edge-weights of cycle i, $1 \leq i \leq d$, are set to $(2 - 2\cos(\frac{2\pi}{n_1}))/(2 - 2\cos(\frac{2\pi}{n_i}))$.*

Other graphs with a similar structure are d-dimensional grids. However, these are not Cayley graphs and it is known that edges of the same dimension do not necessarily need to have the same edge-weight [DMN97]. However, considering them as Cartesian products of paths of length $n_1 \leq n_2 \leq \cdots \leq n_d$, we can also improve the diffusion algorithms on them. Similar to corollary 8, using this approach, the best results are achieved by setting the edge-weight of a dimension i to $(2 - 2\cos(\frac{\pi}{n_1}))/(2 - 2\cos(\frac{\pi}{n_i}))$.

For another example showing the power of this method, consider the Cartesian product of a path of length k with a complete graph of cardinality k^2. Using Theorem 26 and the result of [GMS96], we see that $O(k^4 \cdot \ln(1/\epsilon))$ steps are required to ϵ-balance the system using FOS. Assigning a weight of k^2 to the edges of the path, only $O(k^2 \cdot \ln(1/\epsilon))$ steps are required.

In the following, we consider the Cube Connected Cycles Network of dimension d, which will be denoted by CCC(d). The CCC(d) contains 2^d cycles of length d. We can represent each node by a pair (q, i) where i, $(0 \leq i < d)$ is the position of the node within its cycle and q is a d-bit binary string, where q is the label of the node that corresponds to the cycle. Two nodes (q, i) and (q', i') are adjacent, iff either $q = q'$ and $i - i' = \pm 1 \bmod d$, or $i = i'$ and q differs from q' in exactly the i^{th} bit. Edges of the first type are called *cycle edges*, while edges of the second type are referred to as *hypercube edges*. See Figure 5.1 for an example of CCC(3). Our objective is to determine the edge-weights, for which the diffusion algorithms FOS and SOS will have the fastest convergence. We use the fact that the CCC(d) is a Cayley graph [ABR90]. It is known that the cycles in the CCC(d) are generated by one generator of the corresponding Cayley graph, while the hypercube edges are generated by some other generator. As a consequence of Theorem 25, the CCC(d)'s optimal value for the condition number is obtained, iff all cycle edges are of one weight and all hypercube edges of some other weight. We normalize the weight of the cycle edges to 1 while the weight of the hypercube edges remains variable and is set to a. To compute the optimal value of a we need the following lemmas.

Lemma 15: *Let $C \in \mathbb{R}^{p \times p}$ and $C' = C + a \cdot J$ where $J \in \mathbb{R}^{p \times p}$ with $J_{1,1} = 1$ and all other entries of J equal 0. If we denote with $\lambda_i(C)$ and $\lambda_i(C')$ the i'th eigenvalue of C and C', respectively, then $\lambda_i(C) \leq \lambda_i(C')$ for all $a \geq 0$ and $1 \leq i \leq p$.*

The proof of this lemma immediately follows from the so called Separation theorem [Wil65]. In the next lemma we compute the eigenvalues of a modified Laplacian of a cycle, where one diagonal entry contains the value $2 + 2a$ and all other diagonal entries are set to 2.

Lemma 16: *Let C be the Laplacian of an unweighted cycle of length n and $C' = C + 2a \cdot J$ where $a > 0$ and J is defined as in Lemma 15. Then, $\frac{\lambda_1(C')}{2a+4}$ is maximized for $a = 2 \cdot \sqrt{2}\frac{1}{\sqrt{n}} + O(\frac{1}{n})$.*

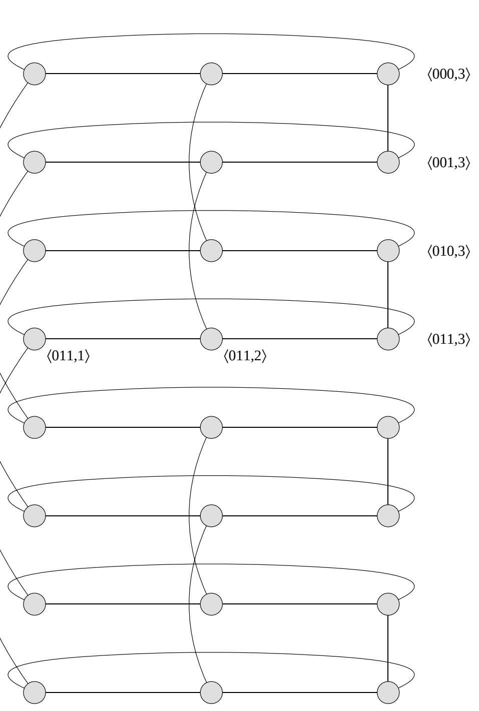


Fig. 5.1: The graph CCC(3). The values in the brackets represent the nodes of the graph.

Proof: Observe, that there exists $C'' \in I\!R^{n\times n}$ such that $C' = 2 \cdot I_n - C''$. First, we compute the characteristic polynomial of C''. Consider

$$\begin{vmatrix} \lambda+2a & -1 & 0 & \cdots & -1 \\ -1 & \lambda & -1 & \cdots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & \lambda & -1 \\ -1 & 0 & \cdots & -1 & \lambda \end{vmatrix} = P_{C_n}(\lambda) + 2aP_{P_{n-1}}(\lambda) = 2\cos(nx) - 2 + 2a\,\frac{\sin(nx)}{\sin(x)},$$

where $P_{C_n}(\lambda)$ is the characteristic polynomial of the cycle of length n, $P_{P_{n-1}}(\lambda)$ is the characteristic polynomial of the path of length $n-1$ and $x = \arccos(\lambda/2)$. Next, we solve the equation $P_{C_n}(\lambda) + 2aP_{P_{n-1}}(\lambda) = 0$.

$$\begin{aligned} P_{C_n}(\lambda) + 2aP_{P_{n-1}}(\lambda) &= 2\cos(nx) - 2 + 2a\frac{\sin(nx)}{\sin(x)} \\ &= 2(\cos^2(\frac{nx}{2}) - \sin^2(\frac{nx}{2}) - \cos^2(\frac{nx}{2}) \\ &\quad - \sin^2(\frac{nx}{2}) + a\frac{2\sin(\frac{nx}{2})\cos(\frac{nx}{2})}{\sin(x)}) \\ &= 4\sin(\frac{nx}{2})(-\sin(\frac{nx}{2}) + a\frac{\cos(\frac{nx}{2})}{\sin(x)}) = 0 \end{aligned}$$

Since $\sin(\frac{nx}{2}) \neq 0$, it holds that $a\cos(\frac{nx}{2}) = \sin(\frac{nx}{2})\sin(x)$. Then we obtain

$$a\cot(\frac{nx}{2}) = \sin(x) \qquad (5.2)$$

The first solution of equation 5.2 satisfies the condition $x = O(1/n)$. Then, $\frac{\lambda_1(C')}{2a+4} = \frac{2-2\cos(x)}{2\sin(x)\tan(\frac{nx}{2})+4} = \frac{1-\cos(x)}{\sin(x)\tan(\frac{nx}{2})+2}$. Denoting with $\rho(x) = \frac{1-\cos(x)}{\sin(x)\tan(\frac{nx}{2})+2}$, we are looking for the first solution of the equation $\rho'(x) = \lim_{y\to x}\frac{\rho(y)-\rho(x)}{y-x} = 0$. Then we obtain

$$\begin{aligned}
\lim_{y\to x}\frac{\rho(y)-\rho(x)}{y-x} &= \lim_{y\to x}\frac{\frac{1-\cos(y)}{\sin(y)\tan(\frac{ny}{2})+2} - \frac{1-\cos(x)}{\sin(x)\tan(\frac{nx}{2})+2}}{y-x} \\
&= \lim_{y\to x}\frac{\frac{y^2-O(y^4)}{(y-O(y^3))\tan(\frac{ny}{2})+2} - \frac{x^2-O(x^4)}{(x-O(x^3))\tan(\frac{nx}{2})+2}}{y-x} \\
&= \lim_{y\to x}\frac{(y^2x\tan(\frac{nx}{2}) + 2y^2 - x^2y\tan(\frac{ny}{2}) - 2x^2) - O(y^4-x^4)}{(y-x)(\sin(y)\tan(\frac{ny}{2})+2)(\sin(x)\tan(\frac{nx}{2})+2)} \\
&= \lim_{y\to x}\frac{xy(y\tan(\frac{nx}{2}) - x\tan(\frac{ny}{2})) + 2(x+y)(x-y) - O(y^4-x^4)}{(y-x)(\sin(y)\tan(\frac{ny}{2})+2)(\sin(x)\tan(\frac{nx}{2})+2)} \\
&= \lim_{y\to x}\frac{xy(y-x)(-x\frac{1}{\cos^2(\frac{nx}{2})} + \tan(\frac{nx}{2})) + 2(y-x)^2 + 4x(y-x)}{(y-x)(\sin(y)\tan(\frac{ny}{2})+2)(\sin(x)\tan(\frac{nx}{2})+2)} \\
&\quad -O(x^3)
\end{aligned}$$

Now we are looking for $\rho'(x) = 0$ and since $4x(y-x) > O(y^4-x^4)$ for large n, it follows that $\tan(\frac{nx}{2}) < x\frac{1}{\cos^2(\frac{nx}{2})}$, which leads to $\frac{\sin(nx)}{2} < x$. Thus, there exists an $\epsilon = O(1/n)$ such that $nx = \pi - 2\epsilon$. Considering equation 5.2 we obtain

$$a(\epsilon + O(\epsilon^3)) = \frac{\pi - 2\epsilon}{n} - O(1/n^3) \Leftrightarrow \epsilon(na+2) = \pi - O(1/n^3) - O(an/n^3)$$

Therefore, $\epsilon = \frac{\pi}{na+2} - O(1/n^3)$. Minimizing $\rho(x)$, we obtain the lemma. □

We are now ready to formulate the following theorem.

Theorem 27: *The optimal value of the condition number of the Laplacian of a weighted CCC(d) is achieved for* $a = 2\cdot\sqrt{2}\frac{1}{\sqrt{d}} + O(\frac{1}{d})$.

Proof: The Laplacian of the weighted CCC(d) is of the form $L_{CCC(d)} = \begin{pmatrix} C_d & D_d \\ D_d & C_d \end{pmatrix}$, where

$$C_1 = L_{C_d} + a\cdot I_d,\ C_k = \begin{pmatrix} C_{k-1} & D_{k-1} \\ D_{k-1} & C_{k-1} \end{pmatrix} \text{ and } D_k = I_{2^{k-1}} \otimes (-a)\cdot J_{(k)}$$

for all $1 \leq k \leq d$. Here, $J_{(k)} \in I\!R^{d\times d}$ with $(J_{(k)})_{d-k+1,d-k+1} = 1$ and all other entries equal 0. C_d represents the unweighted cycle of length d, L_{C_d} its Laplacian and the operation "$\otimes$" is the direct product as defined in Chapter 4: for $A \in I\!R^{m\times n}$, $B \in I\!R^{p\times q}$ the matrix $A \otimes B \in I\!R^{mp\times nq}$ is the matrix obtained from A by replacing every element a_{ij} by the block $a_{ij}B$. The eigenvalues of $L_{CCC(d)}$ are equal to the eigenvalues of the matrices $C_d + D_d$ and $C_d - D_d$. Applying this transformation d times, we obtain some matrices of the form $E_d - A_{C_d}$, where A_{C_d} represents the adjacency matrix of an unweighted cycle of length d. E_d is a diagonal matrix with all diagonal entries belonging to the set $\{2, 2+2a\}$ and all of-diagonal entries are set to 0. Lemma 15 states that the second smallest eigenvalue of the Laplacian of the weighted CCC(d) is the smallest eigenvalue of $E_d - A_{C_d}$, where E_d contains exactly one diagonal entry set to $2+2a$ and all other diagonal entries equal 2. Furthermore, Lemma 15 also implies that the largest eigenvalue of this Laplacian is the largest eigenvalue of $E'_d - A_{C_d}$, where all diagonal entries of E'_d equal $2 + 2a$. Thus, $\rho(x)$ calculated in Lemma 16 equals the condition number of the $L_{CCC(d)}$ and we obtain the theorem. □

Let us now analyze the improvement of the condition number of the Laplacian by setting the weight of the hypercube edges to $\frac{2\sqrt{2}}{\sqrt{d}}$, when $d \to \infty$. We denote with $\varrho(a)$ the quotient between the condition number of the weighted Laplacian (by setting the hypercube edges to a) and the condition number of the unweighted Laplacian. Then it holds

$$\varrho(a) = \left(\frac{2 - 2\cos(\frac{\pi - \frac{2\pi}{da+2}}{d} - O(1/d^3))}{2a+4}\right) \Big/ \left(\frac{2 - 2\cos(\frac{\pi - \frac{2\pi}{d+2}}{d} - O(1/d^3))}{6}\right).$$

This leads to

$$\lim_{d\to\infty} \varrho(a) = \left(\frac{\pi^2}{4} - O(1/\sqrt{d})\right) \Big/ \left(\frac{\pi^2}{6} - O(1/d)\right) = 3/2.$$

Therefore, we can save about 1/3 of the time needed for FOS to balance the load on large topologies of this kind.

For the following, consider the Cube Connected Path, which has a similar structure to the one of the CCC. Its definition is identical to the Cube Connected Cycles, except that the edges between $(q, 0)$ and $(q, d-1)$ are missing. Similar to the CCC, edges of the first type are called *path edges*, while edges of the second type are *hypercube edges*. In the following, we denote the d-dimensional Cube Connected Path of $d \cdot 2^d$ vertices by CCP(d). The CCP is not a Cayley graph and therefore, it is quite difficult to determine optimal parameters for its edges. Anyway, a similar approach can be used to improve the convergence rate, assigning weight 1 to the path edges and a to the hypercube edges. Doing this, the calculations in Lemma 16 and Theorem 27 provide a value of $\frac{\sqrt{2}}{\sqrt{d}} + O(1/d)$ for a. As in the case of the CCC, this value improves the condition number of the Laplacian compared to the unweighted case by a factor of approximately 3/2 for large d.

Other common interconnection topologies are the Butterfly, the wrapped Butterfly, and the de Bruijn graph (see section 4.4 for the definition). Using similar approaches as in the proof of Lemma 16 and Theorem 27, we can improve the polynomial-based load balancing algorithms on these topologies also. However, the improvement factor converges to 1 whenever the cardinality of these graphs converges to ∞ and therefore we do not consider these type of networks in the present section.

5.3 Experimental Results

To show the effects of the approach introduced in Chapter 5.2, a simulation program have been implemented and we have run several tests. Network types included are grid (G), torus (T), Cube Connected Cycles (CCC), Cube Connected Paths (CCP), Butterfly (BF), wrapped Butterfly and de Bruijn (DB). The program was implemented by Stefan Schamberger in C++, using the ARPACK++ library [LSY97] for eigenvalue computations. While it is possible to determine eigenvalues of relatively small networks (e.g. CCC(8)) from the Laplacian itself, we are not able to do this for larger networks (e.g. CCC(16)) in a reasonable amount of time. Therefore, we determine the second smallest and largest eigenvalues of these graphs by either using explicit formulas or by reducing their calculations to the computation of eigenvalues of only parts of the original graph. A detailed description of this approach applied to the CCC can be found in Section 5.2 and we use similar techniques for other hypercubic networks.

Prior to the first iteration of the simulation, the network's load is either distributed randomly (RS) over the network or placed onto a single node (SS), while we normalize the balanced load ($\overline{w_i} = 1$). The total amount of load is therefore equal to the total number of nodes n in the graph. We apply the FOS and the SOS and keep iterating until an almost evenly distributing flow is calculated. For our tests, we define this to be archived as soon as after the k^{th} iteration $|| \, w^k - \overline{w} \, ||_2$ is less than 0.01. For both diffusion schemes, we have chosen the optimal value of $\alpha = \frac{1}{\lambda_2 + \lambda_n}$, for SOS we used $\beta = \frac{2}{1+\sqrt{1-\mu_2^2}}$. The time spent on computing eigenvalues of large graphs is reduced by applying the approach described in Section 5.2, and most of the computation time is consumed by the flow calculations.

Figures 5.2 through 5.7 show some results of our experiments. For each selection of a on the x-axis the resulting convergence rate μ_2 of FOS applied on the specific network type (left) and the number of iterations needed by SOS to compute a balancing flow (right) are shown. Note, that since the results are very similar for any combination of one of the schemes (FOS/SOS) and one of the load patterns (RS/SS), we have only included those for the SOS and SS. The results shown in figures 5.2 through 5.7 are also included

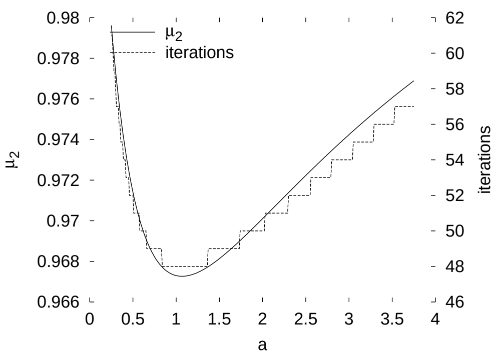


Fig. 5.2: SOS SS CCC(8)

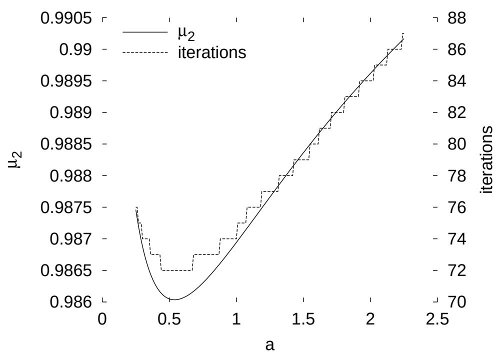


Fig. 5.3: SOS SS CCP(8)

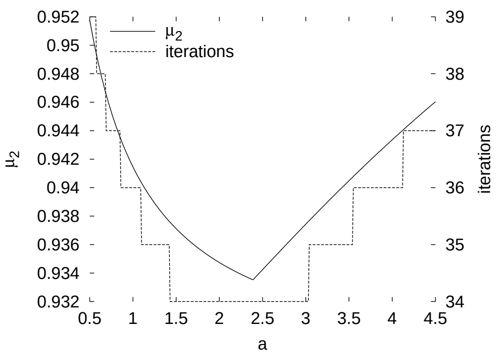


Fig. 5.4: SOS SS wrapped BF(8)

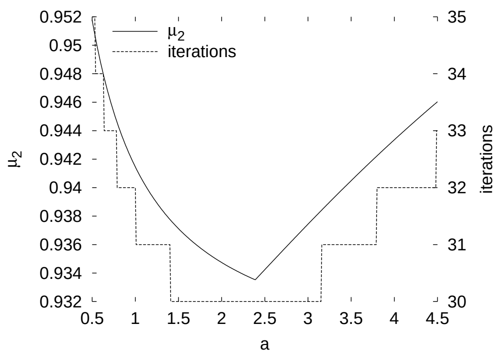


Fig. 5.5: SOS SS DB(8)

in tables 5.2, 5.3 and 5.4, where a short overview on the simulation results with other network sizes is given. As we can see from figures 5.2 to 5.7, the closer a is to the optimal value a_{opt}, the smaller becomes the number of iterations needed to compute a balancing flow on all network types. First, let us study the CCC. In case of the 3 to 8-dimensional CCC we have an optimal a_{opt} greater than 1. We obtain the best improvements for the 4-dimensional CCC and the savings decrease when increasing the dimension. Considering CCC of higher dimensions than 9, we observe that the improvements increase again with larger dimension. As described in Section 5.2, we can win using FOS at most 1/3 for the flow computation when d tends to infinity. Similar savings can be archived for the CCP, but we obtain an a_{opt} value smaller than 1 for the 3-dimensional CCP and the improvements become higher with higher dimensions. Note however, that for the CCC a_{opt} converges to 0 for large n in contrast to wrapped BF and DB, where a_{opt} will stay about the same. A special case is the BF with its optimal value $a_{opt} = 1$. Here, of course, no savings are possible at all, so we omit the corresponding graph. In the case of the wrapped BF and DB, the maximum savings are also modest, ranging from 3% to 14% and as pointed out

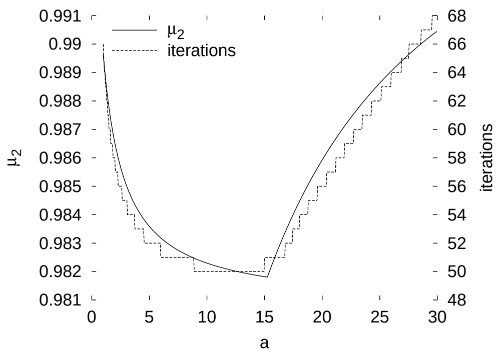


Fig. 5.6: SOS SS G(4 × 16)

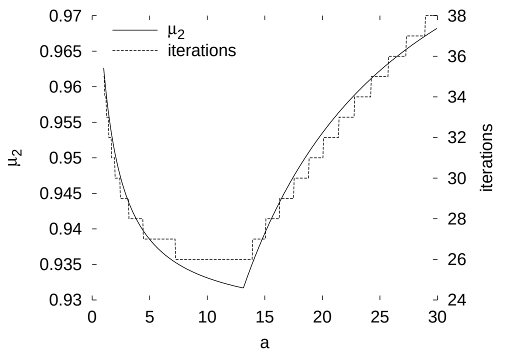


Fig. 5.7: SOS SS T(4 × 16)

in Section 5.2, we cannot expect higher improvements for larger dimensions.

The results for grid and torus given in Figures 5.6 and 5.7 differ from the others in the way that large savings of iterations are possible, what is due to the large value of a_{opt}. As shown in Table 5.3 and 5.4, savings up to 28% can be archived. Note that by fixing one dimension and increasing the other dimension to infinity, the optimal value of a will grow quadratically with the cardinality of the graph in the second dimension leading to improvements up to a factor of 2. We have restricted our experiments to 2-dimensional grid and torus, but similar results can be obtained also for higher dimensional graphs of the same type. This is an interesting result, since these networks of about the same size are widely available. The hpcline [FS] operated by the PC^2 in Paderborn, for example, is designed as an 8 × 12 torus. Hence, these improvements are directly applicable.

	Second Order Scheme (SOS), Single Source (SS)							
n	CCC(n)				CCP(n)			
	Iterations		a_{opt}	Savings	Iterations		a_{opt}	Savings
	$a = 1$	$a = a_{opt}$			$a = 1$	$a = a_{opt}$		
3	16	16	1.50	0%	19	19	0.88	0%
4	23	22	1.50	4%	29	28	0.77	3%
5	28	28	1.29	0%	38	38	0.69	0%
6	35	34	1.23	3%	49	48	0.63	2%
8	48	48	1.07	0%	74	72	0.54	3%
12	83	83	0.87	0%	141	134	0.43	9%
16	127	126	0.75	1%	225	211	0.37	6%

Tab. 5.1: Number of iterations needed to calculate a balancing flow for unweighted CCC and CCP ($a = 1$) and optimal weighted CCC and CCP ($a = a_{opt}$)

n	wrapped BF(n)				DB(n)			
	Iterations		a_{opt}	Savings	Iterations		a_{opt}	Savings
	$a = 1$	$a = a_{opt}$			$a = 1$	$a = a_{opt}$		
3	11	10	2.23	9%	10	9	2.23	10%
4	16	14	2.31	12%	14	12	2.31	14%
5	20	19	2.35	5%	18	16	2.35	11%
6	25	24	2.37	4%	22	21	2.37	5%
8	36	35	2.39	3%	32	30	2.39	6%
12	63	60	2.40	5%	54	52	2.40	4%
16	98	95	2.41	3%	84	81	2.41	4%

Tab. 5.2: Number of iterations needed to calculate a balancing flow for unweighted wBF and DB ($a = 1$) and optimal wBF and DB ($a = a_{opt}$)

Size	Second Order Scheme (SOS), Single Source (SS)							
	grid				torus			
	Iterations		a_{opt}	Savings	Iterations		a_{opt}	Savings
	$a = 1$	$a = a_{opt}$			$a = 1$	$a = a_{opt}$		
4×4	15	15	1.00	0%	9	9	1.00	0%
4×8	31	26	3.85	16%	17	14	3.40	18%
4×12	48	38	8.60	21%	26	20	7.45	23%
4×16	66	51	15.20	23%	35	26	13.10	26%
4×32	137	105	60.80	23%	73	53	52.00	27%

Tab. 5.3: Iterations needed to calculate a balancing flow on a $4 \times x$ grid and torus.

Size	Second Order Scheme (SOS), Single Source (SS)							
	grid				torus			
	Iterations		a_{opt}	Savings	Iterations		a_{opt}	Savings
	$a = 1$	$a = a_{opt}$			$a = 1$	$a = a_{opt}$		
8×8	35	35	1.00	0%	8	8	1.00	0%
8×12	52	45	2.20	13%	27	24	2.20	11%
8×16	71	58	3.95	18%	37	30	3.85	19%
8×32	148	112	15.80	24%	76	57	15.20	25%
8×64	310	228	63.10	26%	159	115	60.80	28%

Tab. 5.4: Iterations needed to calculate a balancing flow on an $8 \times x$ grid and torus.

5.4 Summary

In this chapter, we showed that for edge-transitive graphs the condition number of the corresponding Laplacian is maximized if all edges have the same weight. In section 5.2, we could significantly improve known diffusion load balancing schemes for Cayley graphs and Cartesian products of graphs. Furthermore, we considered known interconnection topologies and showed that, although the benefit is only modest on hypercubic topologies, grid and torus based networks can highly profit by this edge-weighted approach. These results do not only help to improve load balancing software, but can also give valuable information on how to construct communication hardware. Since the amount of load that has to be transferred over a communication edge depends on its type [DFM99], dimensioning the bandwidth accordingly could help to improve performance.

6. DIFFUSION LOAD BALANCING ON HETEROGENEOUS NETWORKS

In chapters 4 and 5, we considered load balancing algorithms in homogeneous processor systems, i.e. each processor of the network had the same computational power and the same amount of memory. In this chapter, we analyze the polynomial-based diffusion algorithms in heterogeneous processor environments.

We show that known diffusion schemes can be generalized for heterogeneous systems and prove that these generalized load balancing algorithms compute an l_2-optimal flow. In Section 6.2.3, we derive relations between the second smallest eigenvalue of the node-weighted Laplacian and the edge expansion of the corresponding heterogeneous network. In order to confirm our theoretical results, we consider several experiments in heterogeneous processor systems with different initial load distribution scenarios.

6.1 Basic Results

Let $G = (V, E)$ be a connected, undirected graph with $|V| = n$ nodes and $|E| = N$ edges. Node $v_i \in V$ has a *weight* of $c_i \in I\!R$ and a *load* of $w_i \in I\!R$. Let $w \in I\!R^n$ be the vector of load values. Our goal is to balance the load proportionally to the weight. We denote with

$$\overline{w} := \frac{\sum_{i=1}^n w_i}{\sum_{i=1}^n c_i}(c_1, \ldots, c_n)^t$$

the vector of a proportionally balanced load. Denote the $n \times n$ diagonal matrix of the weights with C, where the c_i's are the diagonal entries and all other entries are 0.

Let $x \in I\!R^N$ be a flow on the edges of G. We will consider the following generalization of the local iterative load balancing algorithms for node-weighted graphs by illustrating the first order scheme (FOS) of Cybenko [Cyb89]. In each iteration, each node $v_i \in V$ performs local communication with its neighbors only, namely

$$\begin{aligned} \forall e &= \{v_i, v_j\} \in E \text{ perform} \\ y_{i,j}^{k-1} &= \alpha(\frac{w_i^{k-1}}{c_i} - \frac{w_j^{k-1}}{c_j}); \quad x_e^k = x_e^{k-1} + \delta_{i,j} y_{i,j}^{k-1}; \end{aligned} \tag{6.1}$$

and

$$w_i^k = w_i^{k-1} - \sum_{e=\{v_i,v_j\}\in E} y_{i,j}^{k-1} \,. \tag{6.2}$$

As described in Chapter 2, $\delta_{i,j}$ represents the direction of the edge $\{v_i, v_j\}$, $\delta_{i,j} y_{i,j}^k$ is the amount of load sent via edge e in step k. x_e^k is the total load sent via edge e until iteration k and w_i^k is the load of the node i after the k^{th} iteration. The iteration (6.1) can be written in matrix notation as $w^k = Mw^{k-1}$ with the *diffusion matrix* $M = I - \alpha LC^{-1} \in I\!R^{n\times n}$. We will see in Section 6.2.2 that this results in an l_2-minimal flow.

The results of this chapter can also be generalized for edge-weighted graphs, however, here we consider graphs without weighted edges.

In the case of homogeneous networks, the eigenvalues of the positive-semidefinite Laplacian L are used to determine an optimal value for α. Considering heterogeneous networks we have to work with the generalized Laplacian LC^{-1}. Let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of this node weighted Laplacian. Then, the following lemma can be stated.

Lemma 17: *The eigenvalues of the matrix LC^{-1} are real and non-negative. Moreover, they form a basis in $I\!R^n$.*

Proof: First, we prove that LC^{-1} has the same eigenvalues as $C^{-1/2}LC^{-1/2}$. Since it is a real positive-semidefinite symmetric matrix, LC^{-1} has only non-negative real eigenvalues.

Let u be a vector, for which holds $C^{-1/2}LC^{-1/2}u = \lambda u$. Then, $LC^{-1}C^{1/2}u = LC^{-1/2}u = C^{1/2}C^{-1/2}LC^{-1/2}u = \lambda C^{1/2}u$. Thus, $C^{1/2}u$ is an eigenvector of LC^{-1} corresponding to the eigenvalue λ. If we denote with $u_1, \ldots, u_n$ the eigenvectors of $C^{-1/2}LC^{-1/2}$ then for any $x \in I\!R^n$ it holds that $(\frac{1}{\sqrt{c_1}}x_1, \ldots, \frac{1}{\sqrt{c_n}}x_n)^t = \sum_{i=1}^n \eta_i u_i$ for some constants η_i. It follows that $x = \sum_{i=1}^n \eta_i C^{1/2} u_i$. □

If α is chosen appropriately, the diffusion matrix $M = I - \alpha LC^{-1}$ has the distinct eigenvalues $1 = \mu_1 > \mu_2 \geq \cdots \geq \mu_n > -1$ and $(c_1, c_2, \ldots, c_n)^t$ is an eigenvector of M to the simple eigenvalue $\mu_1 = 1$. We define $\gamma = \max\{|\mu_2|, |\mu_n|\} = \max\{|1-\alpha\lambda_2|, |1-\alpha\lambda_n|\}$. A small γ will lead to a fast rate of convergence (Section 6.2.2).

The minimum of γ is achieved for $\alpha = \alpha_{opt} = \frac{2}{\lambda_2+\lambda_n}$, which is the same value as for homogeneous networks w.r.t. the eigenvalues of L. If we use $\alpha = \alpha_{opt}$, it is easy to see that $\gamma = \frac{\lambda_n-\lambda_2}{\lambda_n+\lambda_2} = \frac{1-\rho 1}{1+\rho}$, where ρ is the condition number of LC^{-1}. It shows that a large condition number will lead to a small γ and, therefore, to a fast rate of convergence.

The values of α_{opt} and γ can be calculated for heterogeneous networks in the same way as for homogeneous networks. The only difference is the use of the eigenvalues of LC^{-1}

instead of L. Similar to the homogeneous case, any scheme, for which the work load w^k in step k can be expressed in the form of

$$w^k = p_k(M)w^0 \ , \text{ where } p_k \in \overline{\Pi}_k \ , \tag{6.3}$$

is called a polynomial-based load balancing scheme (as in the case of homogeneous schemes). Again, $\overline{\Pi}_k$ denotes the set of all polynomials p of degree $\deg(p) \leq k$ satisfying the constraint $p(1) = 1$.

Note that the sum over the entries in each column in M equals 1. Furthermore, the condition $p_k(1) = 1$ implies that all column sums in the matrix $p_k(M)$ are 1. Thus, the total work load is conserved, i. e. $\sum_{i=1}^n w_i^k = \sum_{i=1}^n w_i^0$. The representation (6.3) is primarily useful for the mathematical analysis.

The convergence of a polynomial-based scheme will depend on whether (and how fast) the 'error' $e^k = w^k - \overline{w}$ between the iterate $w^k = p_k(M)w^0$ and the balanced load $\overline{w} = \frac{\sum_{i=1}^n w_i^0}{\sum c_i}(c_1, \ldots, c_n)$ will tend to zero. We can generalize Lemma 2 for node weighted graphs in the following way.

Lemma 18: *Let $M = I - \alpha L C^{-1}$, let z_i be the eigenvectors of LC^{-1} and $w^0 = \sum_{i=1}^m \eta_i z_i$ for some constants η_i. Then*

$$\begin{aligned} e^0 &= \sum_{i=2}^n \eta_i z_i, & (6.4)\\ e^k &= p_k(M)e^0, \quad k = 0, 1, 2, \ldots \ . & (6.5) \end{aligned}$$

Proof: Obviously, $\overline{w} = \eta_1 z_1$. The first equality follows from $w^0 = e^0 + \overline{w}$. To show (6.5), we note that due to $p_k(1) = 1$ the vector $\overline{w}$ is an eigenvector of $p_k(M)$ with eigenvalue 1. Thus,

$$e^k = w^k - \overline{w} = p_k(M)(w^0 - \overline{w}) = p_k(M)e^0$$

and the lemma follows. □

6.2 Diffusion Schemes

In this section, we first generalize the load balancing schemes FOS, SOS, CHEBY and OPT for heterogeneous networks and determine their convergence. Then, we show that all generalized schemes compute the unique flow, which is minimal in the l_2-norm. Finally, we discuss the relation between the second smallest eigenvalue and the expansion of the network.

6.2.1 Polynomial Iterative Methods

Using both statements of Lemma 18, we see that the error e^k of any polynomial-based scheme satisfies

$$\begin{aligned} e^k &= p_k(M)\left(\sum_{i=2}^{n}\eta_i z_i\right) = \sum_{i=2}^{n} p_k(M)\eta_i z_i \\ &= \sum_{i=2}^{n} p_k(\mu_i)\eta_i z_i \ . \end{aligned}$$

Here we made use of $p_k(M)\eta_i z_i = p_k(\mu_i)\eta_i z_i$, since z_i is an eigenvector of M with eigenvalue μ_i (z_i is also an eigenvector of LC^{-1}, but to a different eigenvalue). This fundamental relation allows us to analyze several nearest neighbor load balancing schemes in detail. We denote with u_i, $i = 1, \ldots, n$, the eigenvectors of the matrix $C^{-1/2}LC^{-1/2}$. We know that they form a basis, they are orthogonal to each other, and

$$z_i = C^{1/2}u_i.$$

We obtain

$$\begin{aligned} \|e^k\|_2^2 &= \sum_{i=2}^{m} p_k(\mu_i)^2 \|\eta_i z_i\|_2^2 \\ &\leq \left(\|C^{1/2}\|_2^2 \max_{i=2}^{m} p_k(\mu_i)\right)^2 \cdot \sum_{i=2}^{m} \|\eta_i u_i\|_2^2 \\ &\leq \left(\|C^{1/2}\|_2^2 \|C^{-1/2}\|_2^2 p_k(\gamma)\right)^2 \|e^0\|_2^2. \end{aligned}$$

A similar inequality has been obtained in Chapter 2 for homogeneous processor systems.

First-Order-Scheme (FOS)

We start our analysis of different methods with the first order scheme of Cybenko [Cyb89], where one takes $p_k(t) = t^k$. These polynomials satisfy the simple short recurrence $p_k(t) = t \cdot p_{k-1}(t), k = 1, 2, \ldots$, so that we have, with $M = I - \alpha LC^{-1}$,

$$w^k = Mw^{k-1},\ k = 1, 2, \ldots\ .$$

In this situation $|p_k(\mu_i)| = |\mu_i^k| \leq \gamma^k$ for $i = 2, \ldots, m$, where $\gamma = \max_{i=2}^{m} |\mu_i|$. Thus, we obtain

$$\|e^k\|_2 \leq \sqrt{\frac{c_{max}}{c_{min}}}\gamma^k \cdot \|e^0\|_2 \ .$$

Here, c_{max} and c_{min} represents the maximal resp. minimal node weights among the set $\{c_1, \ldots, c_n\}$.

An analogue consideration has been made for reversible ergodic Markov chains, where the convergence of the Markov chain to a stationary distribution has been developed [DS91, SJ89]. The matrix M can be viewed as a transition matrix with the transition probabilities $1/c_j$ from state i to state j and the stationary distribution corresponds to the balanced situation $(c_1, \ldots, c_n)$. The Markov chain is ergodic, because $(M_{ij})c_j = (M_{ji})c_i$ for all $0 \leq i, j \leq n-1$.

Second-Order-Scheme (SOS)

The SOS of [GMS96] takes the polynomials

$$\begin{aligned} p_0 &\equiv 1,\ p_1(t) = t\ , \\ p_k(t) &= \beta t p_{k-1}(t) + (1-\beta) p_{k-2}(t)\ ,\ \ k = 2, 3, \ldots \end{aligned}$$

so that

$$\left.\begin{aligned} w^1 &= Mw^0\ , \\ w^k &= \beta M w^{k-1} + (1-\beta) w^{k-2}\ ,\ \ k = 2, 3, \ldots \end{aligned}\right\} \tag{6.6}$$

Here, β is a fixed parameter. According to the homogeneous case, this iteration converges to $\overline{w}$ whenever $\beta \in (0, 2)$ and the fastest convergence occurs for

$$\beta = \beta_{opt} = 2/\left(1 + \sqrt{1-\gamma^2}\right). \tag{6.7}$$

In this case, we have (see [GV61, Var62])

$$\max_{t \in [-\gamma, \gamma]} |p_k(t)| \ = \ (\beta_{opt} - 1)^{\frac{k}{2}} \left(1 + k\sqrt{1-\gamma^2}\right)\ .$$

Since $\max_{i=2}^{m} |p_k(\mu_i)| \leq \max_{t \in [-\gamma, \gamma]} |p_k(t)|$, we get

$$\|e^k\|_2 \ \leq \ \sqrt{\frac{c_{max}}{c_{min}}} (\beta_{opt} - 1)^{\frac{k}{2}} \left(1 + k\sqrt{1-\gamma^2}\right) \cdot \|e^0\|_2\ . \tag{6.8}$$

According to [GMS96], by comparing the factors γ^k of FOS and $(\beta_{opt} - 1)^{\frac{k}{2}}(1 + k\sqrt{1-\gamma^2})$ of SOS (for γ close to 1), this can be interpreted as SOS being of 'second' order whereas FOS only of 'first' order.

Chebyshev-Scheme (CHEBY)

The Chebyshev method differs from SOS only in the fact that parameter β does now depend on k according to

$$\beta_1 \;=\; 1, \quad \beta_2 = \frac{2}{2-\gamma^2}, \text{ and } \quad \beta_k = \frac{4}{4-\gamma^2\beta_{k-1}}, \tag{6.9}$$

for $k = 3, 4, \ldots$. The corresponding polynomials p_k are the (scaled) Chebyshev polynomials for the interval $[-\gamma, \gamma]$. This means that they are optimal in the sense that (see [GV61, Var62])

$$\begin{aligned} \max_{t\in[-\gamma,\gamma]} |p_k(t)| &= \min_{p\in\overline{\Pi}_k} \max_{t\in[-\gamma,\gamma]} |p(t)| \\ &= (\beta_{opt}-1)^{\frac{k}{2}} \frac{2}{1+(\beta_{opt}-1)^k}. \end{aligned}$$

Similarly to SOS, this yields the estimate

$$\|e^k\|_2 \;\le\; \sqrt{\frac{c_{max}}{c_{min}}}(\beta_{opt}-1)^{\frac{k}{2}} \frac{2}{1+(\beta_{opt}-1)^k} \cdot \|e^0\|_2 \; . \tag{6.10}$$

The factor in (6.10) is always smaller than the factor in (6.8), which shows that the Chebyshev method is usually preferred over the SOS scheme. Asymptotically, however, both methods can be regarded to perform identically since

$$\begin{aligned} \lim_{k\to\infty} \left[\sqrt{\frac{c_{max}}{c_{min}}}(\beta_{opt}-1)^{\frac{k}{2}} \frac{2}{1+(\beta_{opt}-1)^2}\right]^{\frac{1}{k}} &= \\ \lim_{k\to\infty} \left[\sqrt{\frac{c_{max}}{c_{min}}}(\beta_{opt}-1)^{\frac{k}{2}} \left(1+k\sqrt{1-\gamma^2}\right)\right]^{\frac{1}{k}} &= (\beta_{opt}-1)^{\frac{1}{2}} \; . \end{aligned}$$

Optimal-Scheme (OPT)

The knowledge of all distinct eigenvalues of LC^{-1} of the network is needed for OPT [DFM99]. However, it guarantees a completely balanced work load after $m-1$ iterations, where m denotes the number of different eigenvalues of LC^{-1}. Since $m \le n$, the number of iterations does never exceeds the number of nodes in the network. We generalize the simple optimal scheme described in Chapter 4. In the following, we apply the local iterative algorithm with the distinct non-zero eigenvalues $\tilde{\lambda}_k$ of LC^{-1}, $2 \le k \le m$, (in any order) such that

$$\begin{aligned}
&\forall\, e = \{v_i, v_j\} \in E : \\
&\qquad y_{i,j}^{k-1} = \frac{1}{\tilde{\lambda}_k}(w_i^{k-1}/c_i - w_j^{k-1}/c_j); \\
&\qquad x_e^k = x_e^{k-1} + \delta_{i,j} y_{i,j}^{k-1}; \\
\text{and}\quad &\qquad w_i^k = w_i^{k-1} - \sum_{e=\{v_i,v_j\}\in E} y_{i,j}^{k-1}\,.
\end{aligned}$$

Each node i simply adds a flow of $\frac{\delta_{i,j}}{\tilde{\lambda}_k}(w_i^{k-1} - w_j^{k-1})$ to the flow over edge $e = \{i, j\}$ in iteration k, choosing a different eigenvalue for each iteration. Note that $\delta_{i,j}$ represents the direction of the edge. $\delta_{i,j} = 1$ if $i > j$ and $\delta_{i,j} = -1$ otherwise. Therefore,

$$w^k = \left(I - \frac{1}{\tilde{\lambda}_{k+1}} L C^{-1}\right) w^{k-1}\,.$$

After $m-1$ iterations, the error e^{m-1} leads to

$$e^{m-1} = \prod_{i=2}^{m}\left(I - \frac{1}{\tilde{\lambda}_i} L C^{-1}\right)\sum_{j=2}^{n} \eta_j z_j = 0.$$

The order of the eigenvalues can be arbitrarily. However, in Chapter 4 it is shown that one may get trapped in numerically instable conditions and it is also shown how to avoid them.

When comparing the different schemes, SOS is significantly faster than FOS and CHEBY is asymptotically not faster than SOS. All these schemes usually need much more than $m-1$ iterations, whereas OPT does never need more than $m-1$ iterations. Although the calculation of all eigenvalues for OPT can be very time consuming, for a given network we have to compute them only once.

6.2.2 Solution Quality

In this section we consider the quality of the flow computed by the polynomial schemes presented in the previous subsection. The results of [DFM99] obtained for homogeneous networks can also be generalized for heterogeneous networks. In the following, we consider two lemmas from [DFM99], which will be used to prove that the flow is minimal according to the l_2-norm.

Lemma 19: *The equation $Lf = b$ has a solution in f (and then infinitely many), if and only if $b \perp (1, 1, \ldots, 1)^t$.*

We can now state the characterization of l_2-minimal flows.

Lemma 20: *Consider the l_2 minimization problem*

$$\textit{minimize } \|x\|_2 \textit{ over all } x \textit{ with } Ax = b.$$

Provided that $b \perp (1, 1, \ldots, 1)^t$, the solution to this problem is given by

$$x = A^T f, \textit{ where } Lf = b. \tag{6.11}$$

If we solve the equation $Lf = \overline{w} - w^0$ and set $x = A^T f$, then we obtain the l_2-minimal flow for heterogeneous networks. $Lf = \overline{w} - w^0$ has a solution, because, due to the flow conservation property, $\overline{w} - w^0$ is perpendicular to $(1, \ldots, 1)^t$.

In the following lemma we consider iterative methods, where we have a sequence of work loads converging to the average load. For these algorithms the difference vector $\overline{w} - w^0$ does not have to be computed in advance.

Lemma 21: *Let w^k be a (finite or infinite) sequence of work loads converging to the average load $\overline{w}$. Moreover, let*

$$w^k = w^0 + Ax^k$$

be such that $\|x^k\|_2$ is minimal, i. e. (by Lemma 20)

$$x^k = A^T f^k, \textit{ where } Lf^k = w^k - w^0 \ .$$

Then, $\lim_{k \to \infty} x^k = \bar{x}$, $\overline{w} = w^0 + A\bar{x}$, and $\|\bar{x}\|_2$ is minimal.

Proof: Lemma 21 holds for every $\overline{w}$ such that $\overline{w} - w^0$ is perpendicular to $(1, \ldots, 1)^t$. In [DFM99] the lemma was only proved for homogeneous networks, where $\overline{w} = \eta_1 (1, \ldots, 1)^t$. For heterogeneous networks it also holds that $\overline{w} - w^0$ is perpendicular to $(1, \ldots, 1)^t$ and, therefore, the same proof works in this case, too.

Now we are ready to state the main theorem, which shows that the polynomial schemes stated in the previous section provide an l_2-minimal flow.

Theorem 28: *Let $p \in \overline{\Pi}_k$ be a polynomial that satisfies the 3-term recurrence equation*

$$p_k(t) = (\sigma_k t - \tau_k) p_{k-1}(t) + \rho_k p_{k-2}(t) \ , \tag{6.12}$$

with

$$\rho_1 = 0 \textit{ and } \sigma_k - \tau_k + \rho_k = 1 \ \ \forall \ k = 1, 2, \ldots . \tag{6.13}$$

Let

$$d^0 = -\alpha\sigma_1 C^{-1} w^0\,; \quad x^1 = y^0 = A^T d^0\,; \quad w^1 = w^0 + Ay^0\,;$$

and for $k = 2, 3, \ldots$

$$\left.\begin{array}{lcl} d^{k-1} & = & -\alpha\sigma_k C^{-1} w^{k-1} - \rho_k d^{k-2}\,; \\ y^{k-1} & = & A^T d^{k-1}\,; \\ x^k & = & x^{k-1} + y^{k-1}\,; \\ w^k & = & w^{k-1} + Ay^{k-1}\,; \end{array}\right\} \tag{6.14}$$

be the update process for x^k *and* w^k.

Then, $w^k = p_k(I - \alpha LC^{-1})w^0$, $\lim_{k\to\infty} x^k = \bar{x}$ *and* $\lim_{k\to\infty} w^k = \overline{w}$, $\overline{w} = w^0 + A\bar{x}$, $\|\bar{x}\|_2$ *minimal.*

Proof: Using the matrix notation, we obtain for $k = 1$ that $w^1 = w^0 + Ld^0 = p_1(I - \alpha LC^{-1})w^0$.We assume that for any $i < k$, it holds that $w^i = p_i(I - \alpha LC^{-1})w^0$. Then, we get

$$\begin{aligned} w^k &= w^{k-1} + Ay^{k-1} = w^{k-1} + Ld^{k-1} \\ &= w^{k-1} - \alpha\sigma_k LC^{-1}w^{k-1} - \rho_k Ld^{k-2} \\ &= w^{k-1} - \alpha\sigma_k LC^{-1}w^{k-1} - \rho_k(w^{k-1} - w^{k-2}) \\ &= p_k(I - \alpha LC^{-1})w^0 \end{aligned}$$

Using $f_k = \sum_{i=0}^{k-1} d^{k-1}$ and $w^k = w^{k-1} + Ld^{k-1}$ we obtain $Lf^k = w^k - w^0$. Furthermore, $x^k = x^{k-1} + A^T d^{k-1}$ and $x^1 = A^T d^0$ results in $x^k = A^T f^k$. Therefore, the theorem follows from Lemma 21. □

Looking at the schemes discussed so far, we have $\sigma_k = 1$, $\tau_k = \rho_k = 0$ for the FOS so that $d^{k-1} = -\alpha w^{k-1}$. We have $\sigma_k = \beta_k$, $\tau_k = 0$, $\rho_k = (1 - \beta_k)$ in the Chebyshev scheme for each one but the first step. This yields $d^{k-1} = -\alpha\beta_k w^{k-1} - (1-\beta_k)d^{k-2}$. The first step is identical to FOS. The SOS scheme does only differ from Chebyshev by the fact that $\sigma_k = \beta_{opt}$.

6.2.3 The Second Smallest Eigenvalue and the Edge-Expansion

The relation between the second smallest eigenvalue of the Laplacian and the expansion of a graph was discussed in a great detail (see e.g. [Che70, DK86, Moh89]). A similar relation between the Laplace operator of a reversible ergodic Markov chain and the conductance was developed by Sinclair in [Sin93]. Using analog techniques from linear algebra, we provide a relation for the second smallest eigenvalue of LC^{-1} and the edge-expansion of a weighted graph. We define

$$\iota_c(G) = \min_{S \subset V(G)} \frac{E(S, \overline{S})}{\min\{vol(S), vol(\overline{S})\}},$$

where $E(S, \overline{S})$ represents the number of cut edges between S and $\overline{S}$ and $vol(S) = \sum_{i \in S} c_i$. Now we can state the following theorem:

Theorem 29: *For a vertex weighted graph it holds*

$$\frac{\iota_c^2(G)}{2} \min_i \left(\frac{c_i}{deg_i}\right) \leq \lambda_2 \leq 2\iota_c(G) \ ,$$

where λ_2 denotes the second smallest eigenvalue of the matrix LC^{-1} and deg_i represents the vertex degree of $v_i \in V$.

Proof: The second inequality can be proved like in [Chu97]. We concentrate on the first inequality. Let u be the eigenvector of $C^{-1/2}LC^{-1/2}$ corresponding to the second smallest eigenvalue λ_2. We consider the set V_+ of vertices where $z_i > 0$ for $z = C^{-1/2}u$ eigenvector of LC^{-1} corresponding to λ_2. Without loss of generality, we assume that $\sum_{z_i<0} c_i \geq \sum_{z_i \geq 0} c_i$. We define $g(i)$ with $g(i) = z_i$ if $i \in V_+$ and 0 otherwise. We have seen in Lemma 17 that $\lambda_2(LC^{-1}) = \lambda_2(C^{-1/2}LC^{-1/2})$. Then it holds

$$\begin{aligned}
C^{-1/2}LC^{-1/2}u &= \lambda_2 u \\
C^{-1/2}Lz &= \lambda_2 C^{1/2} z \\
Lz &= \lambda_2 C z \\
\sum_{i \in V_+} (Lz)_i z_i &= \lambda_2 \sum_{i \in V_+} (Cz)_i z_i
\end{aligned}$$

Furthermore, we get

$$\begin{aligned}
\sum_{i \in V_+} (Lz)_i z_i &= \sum_{i \in V_+} (deg_i z_i^2 - \sum_{j \in N(i)} z_i z_j) \\
&= \sum_{\{i,j\} \in E(V_+, V_+)} (z_i - z_j)^2 + \sum_{\{i,j\} \in E(V_+, V_-)} z_i(z_i - z_j) \\
&\geq \sum_{\{i,j\} \in E} (g(i) - g(j))^2
\end{aligned}$$

and $\sum_{i \in V_+} (Cz)_i z_i = \sum_{i \in V} g^2(i) c_i$. Relabel the vertices so that $z_i \geq z_{i+1}$, for $1 \leq i \leq n-1$. For every i we consider the cut $C_i = \{\{j,k\} \in E(G) \mid j \leq i < k\}$ and define $\kappa := \min_{1\leq i\leq n} \frac{C_i}{\min\{\sum_{j\leq i} c_j, \sum_{j>i} c_j\}}$. We obtain

$$\begin{aligned}
\lambda_2 &\geq \frac{\sum_{\{i,j\}\in E}(g(i)-g(j))^2}{\sum_{i\in V} g^2(i)c_i} \\
&= \frac{\sum_{\{i,j\}\in E}(g(i)-g(j))^2}{\sum_{i\in V} g^2(i)c_i} \cdot \frac{\sum_{\{i,j\}\in E}(g(i)+g(j))^2}{\sum_{\{i,j\}\in E}(g(i)+g(j))^2} \\
&\geq \frac{(\sum_{\{i,j\}\in E} |g^2(i)-g^2(j)|)^2}{2(\sum_{i\in V} g^2(i)c_i)(\sum_{i\in V} g^2(i)deg_i)} \\
&\geq \frac{(\sum_i |g^2(i)-g^2(i+1)||C_i|)^2}{2(\sum_{i\in V} g^2(i)c_i)^2} \cdot \min_i(\frac{c_i}{deg_i}) \\
&\geq \frac{(\sum_i (g^2(i)-g^2(i+1))\kappa \sum_{j\leq i} c_j)^2}{2(\sum_{i\in V} g^2(i)c_i)^2} \cdot \min_i(\frac{c_i}{deg_i}) \\
&= \frac{\kappa^2}{2}\min_i(\frac{c_i}{deg_i}) \geq \frac{\iota_c^2(G)}{2}\min_i(\frac{c_i}{deg_i})
\end{aligned}$$

and the theorem is proved. □

6.3 Experimental Results

In this section, we present the results of experiments w.r.t. the load balancing schemes described in the previous section. These results are based on experiments executed by Robert Preis. We focus on heterogeneous networks with different weights for the nodes. We choose several networks, each of them consisting of 64 nodes. They are the path of cardinality 64, P_{64}, the square grid of size 8×8, G_8 and the hypercube of dimension 6, $Q(6)$. We experiment with the following heterogeneous node weights.

HOMO: All nodes have the same weight of 1. The total weight is $|V|$. This models a traditional network with homogeneous nodes.

HALF: The first half of nodes has a weight of 2 and the second half of nodes has a weight of 1. The total weight is $\frac{3}{2}|V|$. This models a network, which is a combination of two networks. The nodes of one network are twice as fast as the nodes of the other network.

SERV1: One node has a weight of $|V|+1$ and the others one of 1. The total weight is $2|V|$. This models a network with one server of high weight and with all other nodes of equally small weight. For the P_{64} and the G_8, one node with the smallest degree is chosen as the server node.

Network	Capacity	m	λ_2	λ_n	α_{opt}	β_{opt}	γ
P_{64}	HOMO	64	0.002409	3.997591	0.500000	1.906455	0.998795
	HALF	64	0.001750	3.990781	0.500937	1.919639	0.999123
	SERV1	64	0.001015	3.997553	0.500179	1.938233	0.999492
G_8	HOMO	33	0.152241	7.695518	0.254850	1.567586	0.961201
	HALF	64	0.093933	7.401866	0.266816	1.636018	0.974937
	SERV1	59	0.024937	7.695057	0.259068	1.796160	0.993540
$Q(6)$	HOMO	7	2.000000	12.000000	0.142857	1.176571	0.714286
	HALF	12	1.219224	11.089454	0.162487	1.251980	0.801892
	SERV1	12	0.151868	11.922994	0.165633	1.635482	0.974846

Tab. 6.1: The number n represents the size of LC^{-1} with different weights C. Furthermore, λ_2, λ_n, the optimal values $\alpha_{opt} = \frac{2}{\lambda_2+\lambda_n}$, $\beta_{opt} = \frac{2}{1+\sqrt{1-\gamma^2}}$ and the value of $\gamma = \frac{\lambda_n-\lambda_2}{\lambda_n+\lambda_2}$ are displayed.

Table 6.1 lists the number m of distinct eigenvalues of LC^{-1} for our example networks with different weights C. Additionally, $\tilde{\lambda}_2$ and $\tilde{\lambda}_m$, as well as α_{opt}, β_{opt} and γ are presented. Here, the eigenvalues correspond to LC^{-1}, because we do not consider different values for the edges of the network.

The optimal value α_{opt} is used for the FOS, and the optimal value β_{opt} is used for the SOS. It is always $m = 64$ for the path, because it has a diameter of 63. It is obvious that $m-1$ cannot be less than the diameter for any weight. The network G_8 has a diameter of 14 and m varies between 33 and 64. For the networks G_8 and $Q(6)$, m is always larger for the weights HALF and SERV1 than for HOMO.

The characteristics mentioned so far are independent of the load distribution. We choose the following initial load distribution for our balancing schemes. They model a highly unbalanced as well as a slightly unbalanced load distribution.

PEAK: One node has a load of $100|V|$ and the others have a load of 0.

RAN: $100|V|$ load items are randomly distributed among the nodes.

We compare the FOS, SOS and OPT schemes with each other for various networks and weights. We use the optimal parameter $\alpha_{opt} = \frac{2}{\lambda_2+\lambda_n}$ for the FOS. Additionally, we use the optimal parameter $\beta_{opt} = \frac{2}{1+\sqrt{1-\gamma^2}}$. Both are presented in Table 6.1. Additionally, Table 6.1 lists the value γ, which displays the rate of convergence for FOS and SOS. It can be observed that γ always increases in the following order: HOMO, HALF and SERV1.

We can choose an arbitrary ordering of the eigenvalues for the scheme OPT, but it is possible to get trapped in numerically unstable conditions for certain orderings (as

Network	Load	Capacity	Quality of l_2-minimal flow			iterations		
			$l_1 = \sum \lvert x_e\rvert$	$l_2 = \sqrt{\sum x_e^2}$	$l_\infty = \max\{\lvert x_e\rvert\}$	FOS	SOS	OPT
P_{64}	PEAK	HOMO	201600	29214	6300	9655	294	63
		HALF	167465	25676	6266	13092	340	63
		SERV1	100800	14607	3150	24153	473	63
	RAN	HOMO	1485	253	78	5831	194	63
		HALF	35544	5121	1104	11559	305	63
		SERV1	102211	14732	3147	24174	473	63
G_8	PEAK	HOMO	44800	6849	3150	303	52	27
		HALF	40533	6625	3150	470	65	39
		SERV1	22400	3425	1575	1945	136	33
	RAN	HOMO	573	66	18	189	35	22
		HALF	4749	658	144	339	49	34
		SERV1	22383	3428	1574	1946	136	35
$Q(6)$	PEAK	HOMO	19200	2844	1050	37	18	6
		HALF	18267	2813	1050	56	22	11
		SERV1	9600	1422	525	497	69	11
	RAN	HOMO	416	37	8	23	12	6
		HALF	1450	199	39	38	16	10
		SERV1	9557	1421	526	497	69	11

Tab. 6.2: The l_2-minimal flows and the number of iterations for each scheme.

already mentioned in Chapter 4). We choose the order by the Leja Points [Rei91]. The eigenvalues $\tilde{\lambda}_2, \ldots, \tilde{\lambda}_m$ are sorted to a new sequence $\bar{\lambda}_2, \ldots, \bar{\lambda}_m$, such that $\bar{\lambda}_2 = \tilde{\lambda}_m$ and $\bar{\lambda}_i$, $i = 3, \ldots, m$, with $\prod_{j=2}^{i-1} \left|1 - \frac{\bar{\lambda}_i}{\bar{\lambda}_j}\right| \bar{\lambda}_i$ is maximal.

The balancing process stops after k iterations when the error $||w^k - \overline{w}||_2$ is less than 0.01. Although the optimal scheme OPT results in an error of 0.0 after $m-1$ iterations, the error often gets below 0.01 earlier than that.

As it was demonstrated in the previous section, all schemes calculate the unique l_2-minimal flow. Table 6.2 presents the characteristics of the l_2-minimal flows. The flow for the heavily unbalanced load distribution PEAK is much higher than for the slightly unbalanced load distribution RAN. With respect to the weights, the flow is decreasing in the order HOMO, HALF and SERV1 for the load PEAK and decreasing in the order SERV1, HALF and HOMO for the load RAN. Furthermore, the flow values for the weight SERV1 are almost equal for the load distributions PEAK and RAN. This is true due to the fact that for PEAK, half of the total load has to be moved from the overloaded node to all other nodes. On the other hand, for RAN the node with the highest weight has a load that is about equal to all other nodes and must get about half of the total load from all other nodes.

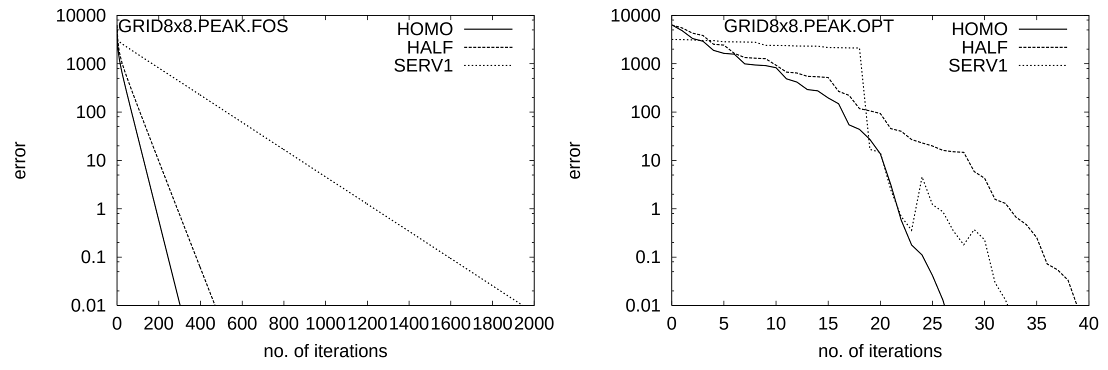


Fig. 6.1: Balancing load PEAK on G_8 with FOS (left) and OPT(right).

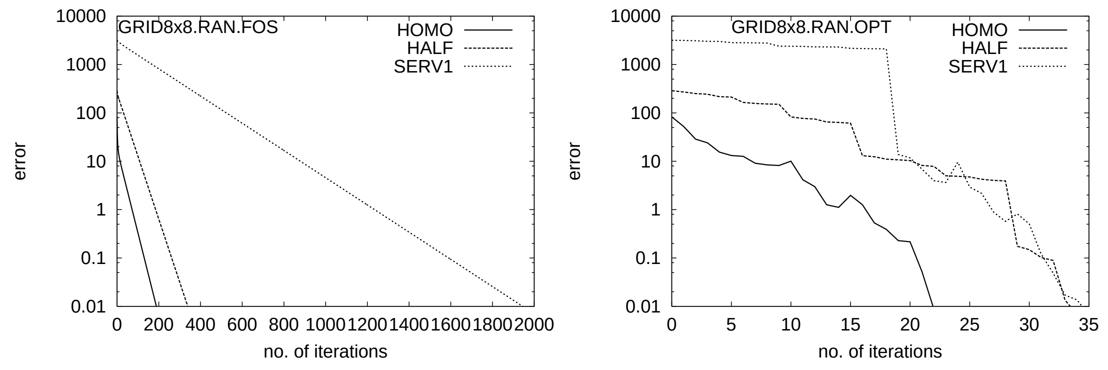


Fig. 6.2: Balancing load RAN on G_8 with FOS (left) and OPT(right).

Table 6.2 lists the number of iterations that are needed for the schemes FOS, SOS and OPT. Obviously, FOS needs many more iterations than SOS and SOS needs more iterations than OPT. For FOS and SOS, the load imbalance is a major factor for the number of iterations, whereas OPT always results in a balanced load after $m-1$ iterations. We can observe that OPT often finishes after less than $m-1$ iterations if the error is already below 0.01.

Figures 6.1 and 6.2 display the relation between the error and the number of iterations for the network G_8. Figure 6.1 presents the load balancing schemes FOS and OPT started with the initial load distribution PEAK and Figure 6.2 presents FOS and OPT started with the initial load distribution RAN.

It is obvious that load balancing on heterogeneous networks can lead to high balancing flows, especially when the weights of the network are very unbalanced. Nevertheless, the algorithms calculate the l_2-minimal flow, which keeps the flow migration low. Furthermore, the number of iterations also increases, but it increases for FOS and SOS much more than for OPT. Additionally, OPT has a maximum number of $m-1 < n$ iterations.

Our experiments show that it is technically easy to modify the existing diffusion load balancing schemes in order to work on heterogeneous networks.

6.4 Summary

In this chapter, we dealt with heterogeneous networks consisting of processors with different computational power and/or with different amount of memory. We generalized the known load balancing diffusion algorithms for these processor networks and showed that that the resulting flow is l_2-minimal. We derived new relations between the node-weighted Laplacian of a graph and its edge-expansion. In Section 6.3, we underlined our theoretical results by several experiments, using heterogeneous processor networks with different initial load distribution scenarios.

7. CONCLUSION

In chapter 3, we analyzed relations between the k-section width of a graph and the first k eigenvalues of its Laplacian. We showed that the classical spectral lower bound w.r.t.the cut size of optimal k-sections is tight only for a very small number of graphs. Moreover, in some cases, a quadratical gap between the classical spectral bound and the k-section width can occur. We introduced the class of level structured graphs and presented a new technique to increase the spectral lower bound, described in inequality (1.1), for this graph class. We were also able to close the previously mentioned quadratic gap for large graphs with a grid-like structure. Furthermore, we analyzed the k-sections of some certain graphs, which do not belong to the class of level structured graphs.

However, these results represent only a small contribution in this field of research. Our objective is to determine, how the structure of a graph influences its spectrum and viceversa. There is plenty of work considering the relations between the second smallest eigenvalue of the Laplacian of a graph and some of its structural characteristics (see e.g. [Alo86, AM85, Alo97, Chu97, HMPR93, AGM87, Moh91c, Moh91b, Moh91a]). Nevertheless, this research area is waiting to be explored.

In Chapter 4, we derived new load balancing algorithms for Cartesian products of graphs, and showed that these algorithms are faster than classical diffusion schemes. We also described a simple optimal scheme and presented families of sparse graphs having a small spectrum.

In Chapter 5, we improved some polynomial-based load balancing algorithms on several graphs, by considering their topological properties. In Chapter 6, we generalized diffusion schemes to heterogeneous processor systems. Furthermore, we stated new spectral bounds w.r.t. the edge-expansion of node-weighted graph.

In some articles, dimension exchange have also been analyzed [Arn01, Arn02, XL97]. It could be shown that these algorithms converge faster than diffusion on some well-known network topologies. We presume that dimension exchange has a better performance than diffusion on any large graph with a bounded vertex degree, but we were not able to prove this conjecture.

Throughout this thesis, we assumed that the processor network is static and synchronous, i.e. it does not change its topology during the computations and it is synchronized by a global clock. In the model of Gosh et al. [GLM$^+$95], asynchronous networks are modelled by dynamic networks. We can generalize the first order scheme to dynamic networks and, using the model of [GLM+95], to asynchronous networks. However, at this moment, we can only guarantee convergence if the dynamicity in the network obey some strong restrictions. Therefore, we intend to analyze if some weaker conditions for the convergence can be determined. Another important question is, if second order schemes also converge under certain conditions in dynamic systems.

LIST OF FIGURES

LIST OF TABLES

BIBLIOGRAPHY

[AAMR93] W. Aiello, B. Awerbuch, B. Maggs, and S. Rao. Approximate load balancing on dynamic and asynchronous networks. In *25th ACM Symp. on Theory of Computing (STOC)*, pages 632–641, 1993.

[ABKU94] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal. Balanced allocations. In *26th ACM Symp. on Theory of Computing (STOC)*, 1994.

[ABR90] F. Annexstein, M. Baumslag, and A. L. Rosenberg. Group action graphs and parallel architectures. *SIAM J. Computing*, 19:544–569, 1990.

[ACMR95] M. Adler, S. Chakrabarti, M. Mitzenmacher, and L. Rasmussen. Parallel randomized load balancing. In *27th ACM Symp. on Theory of Computing (STOC)*, 1995.

[AGM87] N. Alon, Z. Galil, and V.D. Milman. Better expanders and superconcentrators. *Journal of Algorithms*, 8:337–347, 1987.

[AHK87] S.B. Akers, D. Harel, and B. Krishnamurthy. The star graph: An attractive alternative to the n-cube. In *Proc. of the International Conference on Parallel Processing, (ICPP)*, pages 393–400, 1987.

[Alo86] N. Alon. Eigenvalues and expanders. *Combinatorica*, 6(2):83–96, 1986.

[Alo97] N. Alon. On the edge-expansion of graphs. *Combinatorics, Probability and Computing*, 6:145–152, 1997.

[AM85] N. Alon and V.D. Milman. λ_1, isoperimetric inequalities for graphs, and superconcentrators. *J. of Combinatorial Theory*, B(38):73–88, 1985.

[Arn01] H. Arndt. Load balancing by finite dimension exchange algorithms. Technical Report BUGHW-SC 01/4, Bergische Universität GH Wuppertal, 2001.

[Arn02] H. Arndt. On finite dimension exchange algorithms. Technical Report BUGHW-SC 2002/1, Bergische Universität GH Wuppertal, 2002.

[Bah00] J. Bahi. Asynchronous iterative algorithms for nonexpensive linear systems. *J. Parallel Distribut. Comp.*, 60:92–112, 2000.

[BCLS87] T.N. Bui, S. Chaudhuri, F.T. Leighton, and M. Sisper. Graph bisection algorithms with good average case behaviour. *Combinatorica*, 7(2):171–191, 1987.

[BDE99] S.L. Bezrukov, S.K. Das, and R. Elsässer. Optimal cuts for powers of the petersen graph. In *25th International Workshop, WG'99, LNCS 1665*, pages 228–239, 1999.

[BDE00] S.L. Bezrukov, S.K. Das, and R. Elsässer. An edge-isoperimetric problem for powers of the petersen graph. *Annals of Combinatorics*, 4:153–169, 2000.

[BE00] S.L. Bezrukov and R. Elsässer. The spider poset is macaulay. *Journal of Combinatorial Theory*, A 90:1–26, 2000.

[BE01] S.L. Bezrukov and R. Elsässer. Edge-isoperimetric problems for cartesian powers of regular graphs. In *27th International Workshop, WG 2001, LNCS 2204*, pages 9–20, 2001.

[BEM$^+$00] S. Bezrukov, R. Elsässer, B. Monien, R. Preis, and J.-P. Tillich. New spectral lower bounds on the bisection width of graphs. In *26th International Workshop, WG 2000, LNCS 1928*, pages 23–34, 2000.

[Ber00] P. Berenbrink. *Randomized allocation of independent tasks.* Logos-Verlag, 2000.

[BES99a] S.L. Bezrukov, R. Elsässer, and U.-P. Schroeder. On bounds for the k-partitioning of graphs. In *5th Annual Internatinal Conference on Computing and Combinatorics, (COCOON)*, pages 154–163, 1999.

[BES99b] S.L. Bezrukov, R. Elsässer, and U.-P. Schroeder. On k-partitioning of hamming graphs. *Discrete Applied Mathematics*, 95:127–140, 1999.

[Bez96] S.L. Bezrukov. On k-partitioning the n-cube. In *26th International Workshop, WG 1996, LNCS 1197*, pages 44–55, 1996.

[BFG01] P. Berenbrink, T. Fridetzky, and L.A. Goldberg. The natural work-stealing algorithm is stable. In *42th IEEE Symposium on Foundations of Computer Science, (FOCS)*, pages 178–187, 2001.

[Big93] N. Biggs. *Algebraic Graph Theory.* Cambridge University Press, second edition, 1993.

[BJ93] U. Breitschuh and R. Jurisch. *Die Finite-Element-Methode.* Akademie Verlag, 1993.

[BMS97] P. Berenbrink, F. Meyer auf der Heide, and K. Schröder. Allocating weighted jobs in parallel. In *9th ACM Symp. Parallel Algorithms and Architectures (SPAA)*, pages 302–310, 1997.

[Boi90] J.E. Boillat. Load balancing and poisson equation in a graph. *Concurrency - Practice & Experience*, 2:289–313, 1990.

[Bol93] M. Bolla. Spectra, euclidean representations and clusterings of hypergraphs. *Discrete Mathematics*, 117:19–40, 1993.

[BR97] S.L. Bezrukov and B. Rovan. On partitioning grids into equal parts. *Computers and Artif. Intell.*, 16:153–165, 1997.

[BT89] D. Bertsekas and J. Tsitsiklis. *Parallel and Distributed Computation, Numerical Methods.* Prentice-Hall, 1989.

[BT94] M. Bolla and G. Tusnády. Spectra and optimal partitions of weighted graphs. *Discrete Mathematics*, 128:1–20, 1994.

[CDS95] D.M. Cvetkovic, M. Doob, and H. Sachs. *Spectra of Graphs.* Johann Ambrosius Barth, 3rd edition, 1995.

[Che70] J. Cheeger. A lower bound for the smallest eigenvalue of the laplacian. *Problems in analysis*, pages 195–199, 1970.

[Chu97] F.R.K. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional conference series in mathematics.* American Mathematical Society, 1997.

[CY94] F.R.K. Chung and S.-T. Yao. A near optimal algorithm for edge separators. In *Proc. of the 26th ACM Symp. Theory of Computing, (STOC)*, pages 1–8, 1994.

[Cyb89] G. Cybenko. Load balancing for distributed memory multiprocessors. *Journal of Parallel and Distributed Computing*, 7:279–301, 1989.

[Dav79] P.J. Davis. *Circulant matrices.* John and Sons Inc., 1979.

[dBB$^+$97] F. d'Amore, L. Becchetti, S.L. Bezrukov, A. Marchetti-Spaccamela, M. Ottaviani, R. Preis, M. Röttger, and U.-P. Schroeder. On the embedding of refinements of 2-dimensional grids. In *Euro-Par'97 Parallel Processing*, LNCS 1300, pages 950–957, 1997.

[DDLM95] T. Decker, R. Diekmann, R. Lüling, and B. Monien. Towards developing universal dynamic mapping algorithms. In *17th IEEE Symp. Parallel and Distributed Processing (SPDP)*, pages 456–459, 1995.

[DFM99] R. Diekmann, A. Frommer, and B. Monien. Efficient schemes for nearest neighbor load balancing. *Parallel Computing*, 25(7):789–812, 1999.

[DH73] W.E. Donath and A.J. Hoffman. Lower bounds for the partitioning of graphs. *IBM J. Res. Develop.*, 17:420–425, 1973.

[Die98] R. Diekmann. *Load Balancing Strategies for Data Parallel Applications.* Logos-Verlag, 1998.

[DK86] J. Dodziuk and W.S. Kendall. Combinatorial laplacians and isoperimetric inequality. *Pitman Res. Notes Math. Ser.*, pages 68–74, 1986.

[DLMS96] R. Diekmann, R. Lüling, B. Monien, and C. Spräner. Combining helpful sets and parallel simulated annealing for the graph-partitioning problem. *Int. J. Parallel Algorithms and Applications*, 8:61–84, 1996.

[DMM98] R. Diekmann, D. Meyer, and B. Monien. Parallel decomposition of unstructured FEM-meshes. *Concurrency: Practice and Experience*, 10(1):53–72, 1998.

[DMN97] R. Diekmann, S. Muthukrishnan, and M.V. Nayakkankuppam. Engineering diffusive load balancing algorithms using experiments. In *Solving Irregulary Structured Problems in Parallel, (IRREGULAR)*, LNCS 1253, pages 111–122, 1997.

[DMP95] R. Diekmann, B. Monien, and R. Preis. Using helpful sets to improve graph bisections. In D.F. Hsu, A.L. Rosenberg, and D. Sotteau, editors, *Interconnection Networks and Mapping and Scheduling Parallel Computations*, volume 21 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, pages 57–73. AMS, 1995.

[DMP00] T. Decker, B. Monien, and R. Preis. Towards optimal load balancing topologies. In *Euro-Par'00 Parallel Processing*, 2000.

[DS91] P. Diaconis and D. Stroock. Geometric bounds for eigenvalues of markov chains. *The Annals of Applied Probability*, 1:36–61, 1991.

[DSW98] R. Diekmann, F. Schlimbach, and C. Walshaw. Quality balancing for parallel adaptive FEM. In *Solving Irregulary Structured Problems in Parallel, (IRREGULAR)*, LNCS. Springer, 1998.

[DT98] C. Delorme and J.P. Tillich. The spectrum of de bruijn and kautz graphs. *European Journal of Combinatorics*, 19:307–319, 1998.

[EFMP99] R. Elsässer, A. Frommer, B. Monien, and R. Preis. Optimal and alternating-direction loadbalancing schemes. In *EuroPar'99 Parallel Processing*, LNCS 1685, pages 280–290, 1999.

[EKM01] R. Elsässer, R. Královič, and B. Monien. Scalable sparse topologies with small spectrum. In *18th Annual Symposium on Theoretical Aspects of Computer Science, (STACS)*, pages 218–229, 2001.

[ELM01] R. Elsässer, T. Lücking, and B. Monien. New spectral bounds on k-parttioning of graphs. In *13th ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, pages 255–262, 2001.

[EMP00] R. Elsässer, B. Monien, and R. Preis. Diffusive load balancing schemes on heterogeneous networks. In *12th ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, pages 30–38, 2000.

[EMRS02] R. Elsässer, B. Monien, G. Rote, and S. Schamberger. Toward optimal diffusion matrices. In *International Parallel and Distributed Processing Symposium (IPDPS)*, 2002.

[FLS95] C. Farhat, S. Lanteri, and H.D. Simon. TOP/DOMDEC - a software tool for mesh partitioning and parallel processing. *J. of Computing Systems in Engineering*, 6(1):13–26, 1995.

[FMB95] C. Farhat, N. Maman, and G. Brown. Mesh partitioning for implicit computations via iterative domain decomposition. *Int. J. Num. Meth. Engrg.*, 38:989–1000, 1995.

[FOW85] L. Flatto, A.M. Odlyzko, and D.B. Wales. Random shuffles and group representations. *The Annals of Probability*, 13:154–178, 1985.

[Fro90] A. Frommer. *Lösung linearer Gleichungsysteme auf Parallelrechnern.* Friedr. Vieweg & Sohn, 1990.

[FRW94] J. Falkner, F. Rendl, and H. Wolkowicz. A computational study of graph partitioning. *Mathematical Programming*, 66:211–239, 1994.

[FS] Fujitsu-Siemens. hpcline at the *pc*^{2}. http://www.uni-paderborn.de/pc2/services/systems/psc/.

[FS00] A. Frommer and P. Spiteri. On linear asynchronous iterations when the spectral radius of the modulus is one. Technical Report BUGHW-SC 2000/5, Bergische Universität GH Wuppertal, 2000.

[GGL93] A. George, J. R. Gilbert, and J. W. H. Liu. Graph theory and sparse matrix computations. *The IMA Volumes in Math. and its Appl.*, 56, 1993.

[GJ79] M.R. Garey and D.S. Johnson. *COMPUTERS AND INTRACTABILITY - A Guide to the Theory of NP-Completeness*. Freemann, 1979.

[GJS76] M.R. Garey, D.S. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1:237–267, 1976.

[GLM$^+$95] B. Ghosh, F. T. Leighton, B. M. Maggs, S. Muthukrishnan, C. G. Plaxton, R. Rajaraman, A. Richa, R. E. Tarjan, and D. Zuckerman. Tight analyses of two local load balancing algorithms. In *27th ACM Symp. on Theory of Computing (STOC)*, pages 548–558, 1995.

[GM95] S. Guattery and G. Miller. On the performance of spectral graph partitioning methods. In *Proc. of the Sixth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 233–242. ACM-SIAM, 1995.

[GMS96] B. Ghosh, S. Muthukrishnan, and M.H. Schultz. First and second order diffusive methods for rapid, coarse, distributed load balancing. In *8th ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, pages 72–81, 1996.

[GMT95] J.R. Gilbert, G.L. Miller, and S.-H. Teng. Geometric mesh partitioning: Implementation and experiments. In *Proc. Intl. Parallel Processing Symposium (IPPS)*, pages 418–427, 1995.

[Gup96] A. Gupta. WGPP: Watson graph partitioning package. Technical Report RC 20453, IBM Research Report, 1996.

[GV61] G. Golub and R. Varga. Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order richardson iterative methods. In *Numer. Math.*, pages 147–156, 1961.

[Ham92] S. W. Hammond. Mapping unstructured grid computations to massively parallel computers. Technical Report 92.14, NASA Ames Research Center, 1992.

[HB95] Y.F. Hu and R.J. Blake. An optimal dynamic load balancing algorithm. Technical Report DL-P-95-011, Daresbury Lab., UK, 1995.

[HBE98] Y.F. Hu, R.J. Blake, and D.R. Emerson. An optimal migration algorithm for dynamic load balancing. *Concurrency: Practice & Experience*, 10(6):467–483, 1998.

[HL94] B. Hendrickson and R. Leland. The chaco user's guide: Version 2.0. Technical Report SAND94-2692, Sandia National Laboratories, Albuquerque, NM, 1994.

[HL95] B. Hendrickson and R. Leland. An improved spectral graph partitioning algorithm for mapping parallel computations. *SIAM J. Sci. Comput.*, 16(2):452–469, 1995.

[HM92] J. Hromkovič and B. Monien. The bisection problem for graphs of degree 4 (configuring transputer systems). In *Festschrift zum 60. Geburtstag von Günter Hotz*, pages 215–234. Teubner, 1992.

[HM96] V. Heun and E. W. Mayr. Efficient dynamic embedding of arbitrary binary trees into hypercubes. In *3rd Int'l Symposium on Parallel Algorithms for Irregulary Structured Problems (IRREGULAR)*, pages 287–298, 1996.

[HMPR93] C. Helmberg, B. Mohar, S. Poljak, and F. Rendl. A spectral approach to bandwidth and separator problems in graphs. Technical Report 272, Technische Universität Graz, 1993.

[HRW92] S.W. Hadley, F. Rendl, and H. Wolkowicz. A new lower bound via projection for the quadratic assignment problem. *Mathematics of Operations Research*, 17:727–739, 1992.

[Hub75] X.L. Hubaut. Strongly regular graphs. *Discrete Mathematics*, 13:357–381, 1975.

[HW53] A.J. Hoffman and H.W. Wielandt. The variation of the spectrum of a normal matrix. *Duke Math. J.*, 20:37–39, 1953.

[JAMS89] D.S. Johnson, C.R. Aragon, L.A. McGeoch, and C. Schevon. Optimization by simulated annealing: An experimental evaluation; part 1, graph partitioning. *Operations Research*, 37(6):865–893, 1989.

[KL70] B.W. Kernighan and S. Lin. An effective heuristic procedure for partitioning graphs. *The Bell Systems Technical J.*, pages 291–307, 1970.

[Lei92] F.T. Leighton. *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes.* Morgan Kaufmann Publishers, 1992.

[LK87] F.C.H. Lin and R.M. Keller. The gradient model load balancing method. *IEEE Trans. on Software Engineering*, 13:32–38, 1987.

[LM92] R. Lüling and B. Monien. Load balancing for distributed branch and bound algorithms. In *6th Int'l Parallel Processing Symposium (IPPS)*, pages 543–549, 1992.

[LM93] R. Lüling and B. Monien. A dynamic distributed load balancing algorithm with provable good performance. In *5th ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, pages 164–173, 1993.

[LMR91] R. Lüling, B. Monien, and F. Ramme. Load balancing in large networks: A comparative study. In *3rd IEEE Symp. Parallel and Distributed Processing (SPDP)*, pages 686–689, 1991.

[LNRS92] T. Leighton, M. Newman, A. Ranade, and E. Schwabe. Dynamic tree embedding in butterflies and hypercubes. *SIAM Journal on Computing*, 21:639–654, 1992.

[LSY97] R. B. Lehoucq, D. C. Sorensen, and C. Yang. Arpack users' guide: Solution of large scale eigenvalue problems with implicitly restarted arnoldi methods. Technical report, Computational and Applied Mathematics, Rice University, October 1997. Technical Report from http://www.caam.rice.edu/software/ARPACK/.

[MD97] B. Monien and R. Diekmann. A local graph partitioning heuristic meeting bisection bounds. In *8th SIAM Conf. on Parallel Processing for Scientific Computing*, 1997.

[MFKL93] B. Monien, R. Feldmann, R. Klasing, and R. Lüling. Parallel architectures: Design and efficient use. In *10th Annual Symposium on Theoretical Aspects of Computer Science, (STACS)*, pages 247–259, 1993.

[Moh89] B. Mohar. Isoperimetric numbers of graphs. *J. Combin. Theory*, 47(3):274–291, 1989.

[Moh91a] B. Mohar. Eigenvalues, diameter, and mean distance in graphs. *Graphs and Combinatorics*, 7:53–64, 1991.

[Moh91b] B. Mohar. Isoperimetric number of graphs. *Journal of Comb. Theory B*, pages 274–291, 1991.

[Moh91c] B. Mohar. The laplacian spectrum of graphs. *Graph Theory, Combinatorics, and Applications*, pages 871–898, 1991.

[MOW96] F. Meyer auf der Heide, B. Oesterdiekhoff, and R. Wanka. Strongly adaptive token distribution. *Algorithmica*, 15:413–427, 1996.

[MP01] B. Monien and R. Preis. Bisection width of 3- and 4-regular graphs. In *Mathematical Foundations of Computer Science, (MFCS), LNCS 2136*, pages 524–536, 2001.

[MS90] B. Monien and I.H. Sudborough. Embedding one interconnection network in another. *Computing Suppl.*, 7:257–282, 1990.

[OB98] L. Oliker and R. Biswas. Plum: Parallel load balancing for adaptive unstructured meshes. *J. Par. Dist. Comput.*, 52(2):150–177, 1998.

[OD93] S.R. Öhring and S.K. Das. Mapping dynamic data and algorithm structures into product networks. In *Algorithms and Computation, LNCS 762*, pages 147–156, 1993.

[PD97] R. Preis and R. Diekmann. PARTY - A software library for graph partitioning. In *Advances in Computational Mechanics with Parallel and Distributed Processing*, pages 63–71, 1997.

[Pel96] F. Pellegrini. SCOTCH 3.1 user's guide. Technical Report 1137-96, LaBRI, University of Bordeaux, 1996.

[Pre00] R. Preis. *Analyses and Design of Efficient Graph Partitioning Methods.* HNI-Verlagsschriftenreihe, 2000.

[PSL90] A. Pothen, H.D. Simon, and K.P. Liu. Partitioning sparse matrices with eigenvectors of graphs. *SIAM J. on Matrix Analysis and Applications*, 11(3):430–452, 1990.

[PU87] D. Peleg and E. Upfal. The generalized packet routing problem. *Theoretical Computer Science*, 53:281–293, 1987.

[PU89] D. Peleg and E. Upfal. The token distribution problem. *SIAM Journal on Computing*, 18:229–243, 1989.

[Ran91] A. Ranade. Optimal speedup for backtrack search on a butterfly network. In *3rd ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, pages 40–48, 1991.

[Rei91] L. Reichel. The application of leja points to richardson iteration and polynomial preconditioning. *Linear Algebra and its Applications*, 154-156:389–414, 1991.

[Röt98] M. Röttger. *Effiziente Einbettungen von Gittern in Gitter und Hypercubes.* Logos Verlag, 1998.

[RSAU91] L. Rudolph, M. Slivkin-Allalouf, and E. Upfal. A simple load balancing scheme for task allocation in parallel machines. In *3rd ACM Symp. on Parallel Algorithms and Architectures (SPAA)*, pages 237–245, 1991.

[RW95] F. Rendl and H. Wolkowicz. A projection technique for partitioning the nodes of a graph. *Annals of Operations Research*, 58:155–179, 1995.

[SB96] S.A. Savari and D.P. Bertsekas. Finite termination of asynchronous iterative algorithms. *Parallel Computing*, 22:39–56, 1996.

[Sch00] U.-P. Schroeder. *Graph-Einbettungen unter Berücksichtigung von Gitternetzwerken.* PhD Thesis, 2000.

[Sch01] G. Schmidt. *Über die Spektren wichtiger Graphklassen.* Bachelor Thesis, 2001.

[SG99] V.S. Sunderarm and G.A. Geist. Heterogeneous parallel and distributed computing. *Parallel Computing*, 25:1699–1721, 1999.

[Sin93] A. Sinclair. *Algorithms for Random Generation and Counting.* Birkhäuser, 1993.

[SJ89] A. Sinclair and M. Jerrum. Approximate counting, uniform generation and rapidly mixing markov chains. *Inform. and Comput.*, 82:93–133, 1989.

[SKK97] K. Schloegel, G. Karypis, and V. Kumar. Parallel multilevel diffusion schemes for repartitioning of adaptive meshes. In *Proc. EuroPar'97*, LNCS. Springer, 1997.

[ST96] D. A. Spielman and S.-H. Teng. Spectral partitioning works: Planar graphs and finite element meshes. In *Proc. of the 37th Conf. on Foundations of Computer Science, (FOCS)*, pages 96–105, 1996.

[Til99] J.-P. Tillich. The eigenvalues of the double-rooted tree. *Manuscript*, 1999.

[Var62] R. Varga. *Matrix Iterative Analysis.* Prentice-Hall, 1962.

[VFC$^+$96] D. Vanderstraeten, C. Farhat, P.S. Chen, R. Keunings, and O. Zone. A retrofit based methodology for the fast generation and optimization of large-scale mesh partitions: Beyond the minimum interface size criterion. *Comput. Methods Appl. Mech. Engrg.*, 133:25–45, 1996.

[Wal00] C. Walshaw. *The Jostle user manual: Version 2.2.* University of Greenwich, 2000.

[WCDS99] C. Walshaw, M. Cross, R. Diekmann, and F. Schlimbach. Multilevel mesh partitioning for optimising domain shape. *Int. J. High Performance Comput. Appl.*, 13(4):334–353, 1999.

[WCE97] C. Walshaw, M. Cross, and M. Everett. Dynamic load-balancing for parallel adaptive unstructured meshes. In *Proc. 8th SIAM Conf. on Parallel Processing for Scientific Computing*, 1997.

[Wil65] J. H. Wilkinson. *The Algebraic Eigenvalue Problem.* Oxford University Press, 1965.

[XL97] C. Xu and F.C.M. Lau. *Load Balancing in Parallel Computers.* Kluwer, 1997.

[Zie89] O.C. Zienkiewicz. *The finite element method.* McGraw-Hill, 1989.