ELSEVIER

Discrete Applied Mathematics 53 (1994) 211–234

DISCRETE APPLIED MATHEMATICS

Traffic-light scheduling on the grid

Guy Kortsarz, David Peleg[1,*]

Department of Applied Mathematics and Computer Science, The Weizmann Institute, Rehovot 76100, Israel

Received 15 August 1991; revised 5 May 1992

Abstract

This paper studies the problem of route scheduling under the telephone model of communication networks. Previous work in this model considered mostly the "broadcast" and "gossip" communication primitives. The approach studied here is that of devising simple, distributed universal schedules, that are efficient for wide families of routing instances, rather than attempting to solve individual instances separately. The paper concentrates on "traffic-light" type schedules for route scheduling on the two-dimensional grid.

In order to study the problem of scheduling given route instances, routes are classified according to the number of directions they use, and tight bounds are given on the time required for scheduling route instances in each class. For routes of length d or less, using only one direction, scheduling is shown to require $d + O(1)$ time. For simple routes using only two or three directions, scheduling is shown to require $2d + 3$ and $2d + 4$ time, respectively. Finally, for arbitrary simple routes scheduling is shown to require $2d + \Theta(\sqrt{d})$ time.

1. Introduction

The study of useful communication primitives and their efficient implementation is at the heart of current research in the area of communication networks. This paper concerns the common *telephone* communication model (cf. [2]). In this model, messages are exchanged during *calls* placed over edges of the network. A *round* is composed of a collection of calls carried out simultaneously. Each round is assumed to require one unit of time, so round t begins at time $t - 1$ and ends at time t. A vertex may participate in at most one call during a given round, however, there is no limitation on the amount of information that can be exchanged during a given call.

*Corresponding author.

[1]Supported in part by an Allon Fellowship, by a Bantrell Fellowship and by a Walter and Elise Haas Career Development Award.

SSDI 0166-218X(93)E0049-5

Most previous work on communication in the telephone model has concentrated on the two important primitives of *broadcasting* and *gossiping*. A *broadcasting* problem refers to the process of message dissemination whereby a distinguished vertex originates a message that has to be made known to all other processors. A *gossiping* problem refers to the process of performing many broadcasts in parallel, with *each* vertex in the network originating one message. These problems have received considerable attention in the literature; for a comprehensive survey see [2]. In this paper we study the somewhat more generalized *routing* problem, in which each node is allowed to send messages to arbitrary sets of destinations. More specifically, *routing* refers to the process where a number of sender-receiver pairs of vertices are given, and each sender originates a message to be sent to the receiver.

One may distinguish between two versions of the problem. In the more general version, referred to as the *telephone routing problem* (or TR), the algorithm is required to determine both the *routes* along which the messages are to be sent, and the *activation schedules* of the edges, so as to minimize the overall execution time.

This problem is studied, for instance, in [5, 3] on the grid network. A construction given independently in [5] and [3] allows us to solve any TR instance on the grid (that is, the problem of message passing in the grid where the algorithm is required to select the routes) in $d + O(1)$ time units, where d in the maximal distance between an input pair. Furthermore, each message in A that is to be sent from v to w, reaches w in time no larger than the distance from v to w, plus $O(1)$. A scheme with this property is said to enjoy *distance dependent* arrival times.

In this paper we concentrate on a more limited version of the problem, named the *telephone route scheduling problem* (or TRS). In this version, the routes are *given* as part of the input, and it remains only to determine the activation schedules of the edges. We assume that all messages are distinct, but a vertex is allowed to participate in a given instance as a receiver or as a server more than once. For example, a vertex v may send several (different) messages to the same vertex w.

Another distinction of our approach concerns the *nature* of the solutions sought for the problem. Given any particular collection of routing requests, it is possible to look for a set of routes and a schedule that are *optimal* (or near optimal) for that particular instance. This corresponds to the kind of questions studied in several papers dealing with the broadcast and gossip problems. However, this approach typically leads to hard problems. In particular it is shown in [3] that optimizing TR and TRS is NP-hard for general graphs. Restricted graph classes have been given complex centralized solutions [1, 6]. We therefore concentrate on the alternative approach (considered also in [5]) of designing simple, distributed schedules that are *universal* for wide families of routing requests. This naturally leads to the idea of "traffic-light" type schedules, based on *periodic* activation of edges.

Distributed traffic-light solutions are, in particular, suitable for what we call *continuous* routing problems. In the TRS problem, there is a distinguished starting

point in time, upon which the input is presented to the processors and all the vertices start sending their messages. In contrast, a continuous TRS problem (denoted CTRS) is a version of the problem in which there is no such starting point. Rather, the system operates continuously, and at any given moment, any vertex may initiate a message to any other vertex on some specific path. The solution to such a problem is an algorithm that minimizes the time duration since a message is introduced into the system, until it is delivered at its destination. The solutions given in this paper for the TRS problem are especially oriented for continuous problems.

Let us now briefly describe the results presented in this paper. In Section 2.3 we define the notion of *traffic-light* schedule. We then turn our attention to studying the problem of scheduling given route instances on the grid. It turns out that in order to analyze the resulting complexities, it is convenient to classify the routes according to the number of *directions* they use. Using this classification, it is possible to establish tight bounds on the time required for scheduling route instances in each class.

In Section 3 we deal with the case where the input routes are one-directional. We give a (surprisingly nontrivial) single schedule S_2 that completes the routing on any such problem instance A, in no more than $d(A) + O(1)$ time units, where $d(A)$ is the maximal length of an input route in the input. (It is worth noting that the naive Manhattan-type approach would schedule in $d + O(1)$ time only those routes that go "in the right direction" of their row or column; routes going in the wrong direction would require $3d$ time units.)

The arrival times of the messages under schedule S_2 are distance dependent. An application of this schedule to the classical broadcasting and gossiping problems, establishes that it is possible to achieve broadcast and gossip on the grid with distance-dependent, O(1) deviation from the optimum time, using *shortest paths* between every sender-receiver pair.

In Section 4 we deal with TRS on the grid when the input routes each use only 2 or 3 directions. In Section 4.1 we present the traffic-light schedule S_3, and show that this schedule completes the routing on every such instance A in no more than $2 \cdot d(A) + O(1)$ time units. The additive constant is 3 if the routes in A are shortest paths, i.e. 2-directional, and 4 if the routes in A are 3-directional. In Section 4.2 we study some aspects of TR on trees, and we use these results in Section 4.3 to prove the optimality of schedule S_3 in the worst case, by constructing a 2-directional problem instance P_1 and a 3-directional problem instance P_2 that require at least these times under any schedule.

Finally, in Section 5 we deal with the TRS problem on the grid where the input routes are arbitrary (4-directional) simple paths. In Section 5.1 we analyze the performance of the S_3 traffic-light schedule on such routes. We show that S_3 completes the routing in any such problem A in no more than $2 \cdot d(A) + 2\sqrt{2} \cdot \sqrt{d(A)} + 2$ time units. This bound is complemented in Section 5.2, where we construct a problem A for which the time needed to complete the routing is at least $2 \cdot d(A) + 2 \cdot \sqrt{d(A) + 1} - 1$.

2. Preliminaries

2.1. The model

The communication network is modeled by a connected graph $G = (V, E)$ consisting of a set V of n vertices, $V = \{v_1, \ldots, v_n\}$ representing the processors, and a set E of m edges, $E = \{e_1, \ldots, e_m\}$ representing the communication lines between the processors.

In this paper we focus on the (rectangular) $m \times n$ *grid*, which is a graph $\mathrm{Gr}_{m,n} = (V, E)$ defined as follows:

$$V = \{(i,j)\colon 1 \leqslant i \leqslant m,\ 1 \leqslant j \leqslant n\},$$

$$E = \{((i,j),(i,j+1))\colon 1 \leqslant i \leqslant m,\ 1 \leqslant j \leqslant n-1\}$$

$$\cup\, \{((i,j),(i+1,j))\colon 1 \leqslant i \leqslant m-1,\ 1 \leqslant j \leqslant n\}.$$

The path consisting of vertices with a fixed first (respectively, second) index i is called *the ith row* (resp., column). We think of the grid as embedded in the plane so that $(1, n)$ is at the top, rightmost corner. Given a route (i.e., an oriented path) $\rho = (e_1, e_2, \ldots, e_{|\rho|})$ on the grid, we associate with each move e in ρ a *direction*, $\mathrm{dir}(e) \in \{\text{"l"}, \text{"r"}, \text{"u"}, \text{"d"}\}$, in the obvious way. We may also refer to a move $e = (i, j+1) \to (i, j)$ on the route (i.e., such that $\mathrm{dir}(e) = \text{"l"}$) as an "l" *move*, and say that the route is moving in the left direction, and similarly for the other directions.

Denote the maximal degree of a vertex in the network $G = (V, E)$ by $\Delta(G)$ (or simply Δ, if the context is clear). We denote a path ρ by $\rho = (e_1, e_2, \ldots, e_{|\rho|})$ where e_i are the path edges, however, we sometimes denote $\rho = (u_0, u_1, \ldots, u_{|\rho|})$, where $e_i = (u_{i-1}, u_i)$. The prefix containing the first i edges of the route is denoted by $\rho(i) = (e_1, \ldots, e_i)$, for all $1 \leqslant i \leqslant l$. An edge e on the route is regarded as a *directed* edge, i.e., an edge with the orientation imposed by the route. We shall refer to the edge e_i as the ith *move* on the route.

Given a graph $G = (V, E)$, two edges $e_1, e_2 \in E$ are called *adjacent* if they share a vertex. A subset of edges $E' \in E$ is an *independent set* (of edges) iff every two edges $e_1, e_2 \in E'$ are nonadjacent.

Throughout the sequel we use the following terminology to describe the behavior of the network during the execution of a communication protocol. At a given round, if a call is placed over an edge e we say that the edge e is *active* in this round, else we say that the edge is *idle*. Note that by the definition of the telephone model, at each round t, the set of active edges must be independent. A *schedule* on a graph $G = (V, E)$ is an infinite sequence $(E_i)_{i \geqslant 1}$ such that for each i, $E_i \subset E$ and E_i is independent. Intuitively, a schedule determines the set E_i of active edges on the ith round. A schedule induces for every edge e a (possibly infinite) tuple $\mathscr{A}(e) = (t_e^1, t_e^2, \ldots)$, consisting of the rounds in which the edge e is active in the schedule. Thus the collection $\{\mathscr{A}(e) \mid e \in E\}$ completely characterizes the schedule.

In order to simplify the process of designing and analyzing our schedules, it is sometimes convenient to view every edge e as two directed edges $\vec{e}$ and $\overleftarrow{e}$ in opposite directions, and treat these edges separately. In particular, $\vec{e}$ and $\overleftarrow{e}$ may be given separate sets $\mathscr{A}(\vec{e})$ and $\mathscr{A}(\overleftarrow{e})$ of active rounds (these sets may of course intersect). Note that such a *directed schedule* must still obey the rules of the model. It can be thought of as obtained by starting with a regular bidirectional schedule (in which, whenever an edge e is active, it can be used in both directions), and restricting it by forbidding the use of some edges e in one of the directions at some of their active rounds. While such restrictions yield no actual gains for us (and in fact, can only degrade the performance of the resulting algorithm), they serve to simplify the analysis in some cases. When dealing with a directed schedule, we denote the directed graph obtained from G by splitting the edges as described by $\hat{G}$.

2.2. The TRS problem

Posed formally, a TRS problem is defined as follows. The input consists of a set of k routes, $\{\rho_1, \ldots, \rho_k\}$. For every ρ_i, its start vertex v_i originates a message M_i, to be sent to its end vertex w_i. Each v_i is called a *sender* and each w_i is called a *receiver*. Any vertex v may appear more than once as a server and as a receiver. The goal is for each message M_i to be sent to w_i along ρ_i, where the communication pattern obeys the requirements of the telephone model. A solution for an instance of the TRS problem consists of a schedule, that is, the solution must determine the (independent) set of edges E_t to be active at round t, for every $t \geqslant 1$. Such a schedule completely specifies the progress of messages for the TRS instance, that is, for every path, the vertex currently holding the message forwards it to the next vertex in the path, in the next time that the corresponding edge is active. Given an instance A of the TRS problem, we denote $d(A) = \max_i\{|\rho_i|\}$. Denote the time needed to complete the routing in A by $T(A)$. Note that $T(A) \geqslant d(A)$ for every TRS instance A.

Given a nonnegative integer d, denote the family of TRS problem instances A such that $d(A) \leqslant d$ by TRS_d. We measure the efficiency of an algorithm P for the set TRS_d by the number of time units, $T(P, d)$, required to complete the routing in the *worst case*, using the algorithm. We denote by TRS(i-DIR), $i \in \{1, 2, 3, 4\}$, the TRS problem on the grid $\mathrm{Gr}_{m,n}$ where the input routes are i-directional (that is, the routes use only i of the four possible forwarding directions on the grid).

2.3. Traffic-light schedules

In order to cope with the TRS problem, a possible approach is to ignore the particular inputs, and try to give a single *universal schedule* that deals simultaneously with a large set of problem instances. This motivates the following definition.

Definition 2.1. (1) A *traffic-light* schedule for a family $\mathscr{S}$ of TRS instances on a *fixed* graph G is a *fixed* schedule (i.e., a schedule that does not depend on the particular

instance) such that for every edge e, if $\mathscr{A}(e) \neq \emptyset$ then it is *periodical*. Denote the period of a nonempty sequence $\mathscr{A}(e)$ by $\bar{p}(e)$. We define the period $\bar{p}$ of the traffic-light schedule to be the least common multiplier of the periods $\bar{p}(e)$ of all the edges $e \in E$ with nonempty $\mathscr{A}(e)$.

(2) A fixed schedule for a family S of TRS instances said to be a *directed traffic-light schedule* if it is a traffic-light schedule defined on $\hat{G}$, the directed version of G, that is, both $\mathscr{A}(\vec{e})$ and $\mathscr{A}(\overleftarrow{e})$ are either empty or periodical. We define the period of the directed traffic-light schedule to be the least common multiplier of all the periods of all nonempty $\mathscr{A}(\vec{e})$ and $\mathscr{A}(\overleftarrow{e})$ for all $e \in E$.

The next theorem follows from [5, 3], using traffic-light schedules.

Theorem 2.2. *For any TRS instance A, on an arbitrary graph G, there exists a single traffic-light schedule that completes the schedule on A within $\Delta(G) \cdot d(A) + 1$ time units.*

We comment that any traffic-light algorithm can be used to solve the CTRS problems as well. In fact, all the algorithms given in the next sections are suited for such problems.

3. Scheduling 1-directional paths on the grid

In this section we study scheduling of one-directional routes. Every vertex is allowed to send a message only to vertices in the same row or column, along this row (column). The "Manhattan-type" schedule used in [5, 3] can, in principle, be used here. However, in every row or column, this schedule allows delay-free message propagation only in one direction (by providing a "green-wave" in that direction). Thus if an input route requires the message to be sent in the direction opposite to the "green-wave", it leads to a $3 \cdot d + O(1)$ scheduling time. In this section we present a solution for this problem that achieves $d + O(1)$ scheduling time, which is optimal up to an additive constant. Intuitively, we achieve this by making sure that each row and column in the grid provides a "green wave" in *both* directions simultaneously. This seemingly simple requirement turns out to be surprisingly nontrivial to obtain.

The input of an instance in TRS(1-DIR) contains four kinds of sender-receiver routes:

- (i, j) sends a message to (i, k) along the ith row using only "l" moves if $j > k$, and only "r" moves if $k > j$.
- (i, j) sends a message to (k, j) along the jth column using only "d" moves if $k > i$ and only "u" moves if $i > k$.

Since the schedule is directed we look at each edge $e = ((i, j), (i, j + 1))$ as two directed edges $e_l = \langle (i, j + 1), (i, j) \rangle$ and $e_r = \langle (i, j), (i, j + 1) \rangle$ in two directions. Similarly a vertical edge $e = ((i, j), (i + 1, j))$ is viewed as two edges $e_d = \langle (i, j), (i + 1, j) \rangle$ and $e_u = \langle (i + 1, j), (i, j) \rangle$.

Definition 3.1. A directed traffic-light schedule for the $m \times n$ grid $\mathrm{Gr}_{m,n}$ is *directed rightward-wave* on the ith row, if

(1) For each round $t \geqslant 1$ and $1 \leqslant j \leqslant n-1$, if the edge $\langle (i, j-1), (i, j) \rangle$ is active on round t, then the edge $\langle (i, j+k-1), (i, j+k) \rangle$ is active on round $t+k$, for all $1 \leqslant k \leqslant n-j$.

(2) For every edge e on row i, $\mathscr{A}(\check{e}), \mathscr{A}(\hat{e}) \neq \emptyset$.

Definition 3.2. A traffic-light schedule for a given grid $\mathrm{Gr}_{m,n}$ is called *bi-wave* if it is *rightward-directed wave* and *leftward-directed wave* on each row, as well as *upward-directed wave* and *downward-directed wave* on each column.

Note that if a schedule is rightward-directed wave on a row, for example, then once a message leaves its origin in that row, it can be sent rightward without delaying in the way. We now describe a directed bi-wave traffic-light schedule for the grid $\mathrm{Gr}_{m,n}$. Such a schedule will allow each message to wait only O(1) time units before departing, and never stop again. This of course leads to distance-dependent O(1) delay of each message. Intuitively, we want to take advantage of the wave property of the schedule by making the message join a wave in the appropriate direction and "ride" it all the way to the destination, progressing by one row or column at each step. All a message has to do is to wait for the wave to reach its source and then start marching with it. The active times for any possible edge is described in the following schedule. For any column index i we define two numbers $d_u(i)$ and $d_d(i)$ depending only upon i mod 24 (see Fig. 1).

Algorithm 3.3. *Directed bi-wave traffic-light schedule* S_2

(1) Let e be an horizontal edge $e = ((i, j), i, j+1))$,

$$\mathscr{A}(e_r) = \{t > 0 \colon t \equiv (j-k) \bmod 12\},$$

$$\mathscr{A}(e_l) = \{t > 0 \colon t \equiv (q-j) \bmod 8\},$$

12	11	10	9	8	7	6	5	4	3	2	1	i mod 24
1	2	2	11	12	21	7	8	8	17	3	4	"d"
15	0	0	1	10	11	21	6	6	7	17	2	"u"

24	23	22	21	20	19	18	17	16	15	14	13	i mod 24
4	14	14	23	9	10	10	19	5	6	6	15	"d"
2	12	12	13	23	8	8	9	19	4	4	5	"u"

Fig. 1. Definition of $d_d(i)$ and $d_u(i)$ for $1 \leqslant i \leqslant 24$.

where

$$k = \begin{cases} 0, & \lfloor (i-1)/6 \rfloor \text{ is even,} \\ 6, & \text{otherwise;} \end{cases}$$

$$q = \begin{cases} 2, & \lfloor (i-1)/4 \rfloor \text{ is even,} \\ 6, & \text{otherwise.} \end{cases}$$

(2) Let e be a vertical edge $e = ((i,j),(i+1,j))$,

$$\mathscr{A}(e_d) = \{t > 0 \colon t \equiv (d_d(j) + i) \bmod 24\},$$

$$\mathscr{A}(e_u) = \{t > 0 \colon t \equiv (d_u(j) + (6 - i)) \bmod 24\}.$$

In order to prove that this traffic-light schedule obeys the rules of the telephone model, we must show that there are no two adjacent edges that are active at the same round. This property can be verified by a tedious but straightforward case analysis. A different proof is presented in [3], based on considering the scheduling of arbitrary messages, and showing that for an arbitrary input, two given messages of arbitrary types are not being delivered in two adjacent edges at the same round.

We note again that since the schedule is *bi-wave*, if v sends a message to w along a 1-directional route, it arrives in $\mathrm{dist}(v, w) + O(1)$ time units (where $\mathrm{dist}(v, w)$ is the distance of v and w in the grid); the message may initially be delayed for up to 23 rounds, and is no longer delayed afterward.

Theorem 3.4. *For every problem instance* $A \in \mathrm{TRS}(1\text{-DIR})$, *schedule* S_2 *guarantees that* $T(A) = d(A) + O(1)$.

The schedule S_2 has an important application to the problem of broadcasting and gossiping in the grid. Farley and Proskurowski [1] give an optimal algorithm for the gossiping problem in the grid. Using the traffic-light algorithm of [5, 3] instead (called S_1 in [3]), the gossip is completed in OPT + 9 time units (where OPT denotes the minimum possible time). Moreover, this "traffic-light" method has the additional attractive feature of *distance-dependent* arrival times, that is, by using S_1 a message is delivered from v to w in no more than $\mathrm{dist}(v, w) + 9$ time units. Note, however, that the messages are not always sent in S_1 along shortest paths. This can be remedied using the schedule S_2 presented above. Using this schedule it is possible to perform gossip in such a way that the message sent from v and w, will be transmitted along a shortest path, and will arrive after no more than $\mathrm{dist}(v, w) + 35$ time units. For example, if (i, j) wants to send a message to (k, l), first route the message to (k, j) and then to (k, l) (using S_2).

Theorem 3.5. *It is possible to perform broadcast and gossip on the grid along shortest paths, with only* $O(1)$ *distance-dependent delay.*

4. Scheduling shortest paths and 3-directional paths on the grid

In this section we are interested in the problems TRS(2-DIR) and TRS(3-DIR), namely, route scheduling on the grid $\mathrm{Gr}_{m,n}$, in the special case where the input routes are shortest paths or 3-directional paths. In Section 4.1 we present the "traffic-light" schedule S_3 for this class of instances, achieving an upper bound on the schedule time. In Section 4.2 we study some properties of TRS on trees, which are then used in Section 4.3 to construct a matching lower bound.

4.1. The upper bound

The solution for TRS(2-DIR) and TRS(3-DIR) is based on a new schedule, S_3, which we shall call the *cyclic* schedule. The schedule S_3 relies on a partition of the grid to 3×3 blocks. Each edge is active once every four rounds. In Fig. 2 we give the schedule of the first five rows and columns. The number beside each edge denotes the times when the edge is active (modulo 4).

The name for this schedule stems from the fact that rather than providing a "green wave" along rows and columns, it enables a wave-like progress for *counterclockwise movements* on each internal face of the grid.

In order to analyze the time it takes to complete the routing along a given path using the schedule we give the following definition.

Definition 4.1. We say that a given route $\rho = (e_1, \ldots, e_l)$ takes a *clockwise* (respectively, *counterclockwise*) *turn* in the grid, when it makes two consecutive moves $\{e_i, e_{i+1}\}$ of the types depicted in Fig. 3(a) (resp., Fig 3(b)).

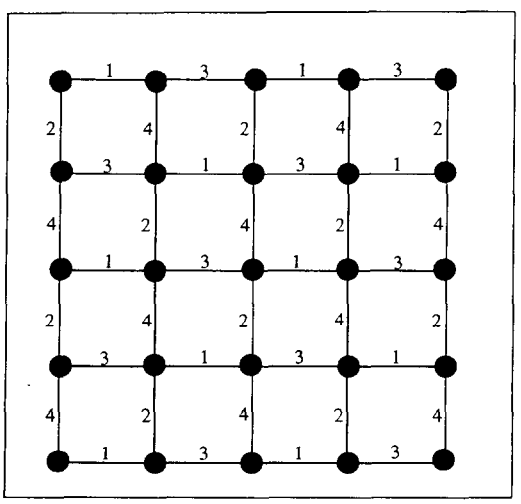

Fig. 2. Active times for edges in the cyclic traffic-light schedule S_3.

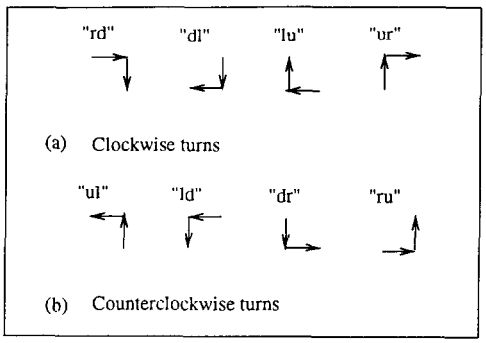

Fig. 3. Clockwise and counterclockwise turns in the grid.

We refer to a clockwise turn that uses the directions "r" and "d", as a "rd" turn, and analogously for the other types (see Fig. 3). Let us define the function "*next*" on clockwise turns by setting next("ur") = "rd", next("rd") = "dl", next("dl") = "lu" and next("lu") = "ur".

Given a route ρ on the grid, let $\omega(\rho)$ denote the difference between the number of clockwise turns in ρ and the number of counterclockwise turns in ρ. The following fact can be observed directly from the definition of schedule S_3.

Fact 4.2. *Given a route ρ, the number of time units it takes to traverse ρ using the cyclic traffic-light schedule S_3 is at most $2 \cdot |\rho| + \omega(\rho) + 2$.*

It follows from Fact 4.2 that in order to bound the routing completion time under schedule S_3 it is necessary to analyze $\omega(\rho)$ for a given route ρ. This analysis is based on looking at the sequence of "significant" points along the run. These points are the ones in which $\omega(\rho)$ increases by one, and never falls below the current value afterwards. Bounding the number of such "significant" points in time immediately gives an upper bound on $\omega(\rho)$. Formally, we introduce the following definition.

Definition 4.3. The "tuple of significant turns" of a route ρ is the tuple of integers $S(\rho) = (t_1, \ldots, t_k)$ with the following properties. (Recall that $\rho(i)$ denotes the i-prefix of ρ.)

(1) $1 \leqslant t_1 \leqslant \cdots \leqslant t_k \leqslant |\rho|$,
(2) $\omega(\rho(t_i)) = i$, for $1 \leqslant i \leqslant k$,
(3) for every $1 \leqslant i \leqslant k$ and $t_i \leqslant t \leqslant |\rho|$, $\omega(\rho(t)) \geqslant i$,
(4) for every $1 \leqslant i \leqslant k$, t_i is the least integer satisfying properties (2) and (3).

The tuple $S(\rho)$ of significant turns is said to be of size $|S(\rho)| = k$. In case such a tuple does not exist (e.g., when $\omega(\rho) \leqslant 0$), $S(\rho)$ is empty and $|S(\rho)| = 0$.

Fact 4.4. *For every route ρ, $\omega(\rho) \leqslant |S(\rho)|$.*

Note that properties (3) and (4) of Definition 4.3 guarantee that if t_i is an element in $S(\rho)$, then ρ takes a clockwise turn in its $(t_i - 1)$st and t_ith moves. We now claim:

Claim 4.5. *Suppose that t_i and t_{i+1} are two consecutive entries in the tuple $S(\rho)$ of significant turns of a route $\rho = (e_1, \ldots, e_{|\rho|})$, and denote the turns taken by ρ in its moves $\{e_{t_i-1}, e_{t_i}\}$, and $\{e_{t_{i+1}-1}, e_{t_{i+1}}\}$ by T_{t_i} and $T_{t_{i+1}}$ respectively. Then $T_{t_{i+1}} = \text{next}(T_{t_i})$.*

Proof. By the definition of significant turns, $\omega(\rho_{t_i}) = \omega(\rho_{t_{i+1}-1}) = i$ and $\omega(\rho_{t_{i+1}}) = i + 1$. Thus in the segment of the route from move e_{t_i+1} until move $e_{t_{i+1}-1}$, there is an equal number of clockwise and counterclockwise turns, and move $e_{t_{i+1}-1}$ takes one more clockwise turn. It follows that moves e_{t_i} and $e_{t_{i+1}-1}$ take the same direction, and the claim follows. □

Lemma 4.6. *Given an i-directional route ρ on a grid, for $i \leqslant 3$, ρ's tuple of significant turns satisfies $|S(\rho)| \leqslant i - 1$.*

Proof. Suppose $S(\rho)$ is nonempty, and let $S(\rho) = (t_1, t_2, t_3, \ldots)$. Assume without loss of generality that $\text{dir}(e_{t_1-1}) =$ "l". Then by Claim 4.5, $\text{dir}(e_{t_1}) =$ "u", $\text{dir}(e_{t_2}) =$ "r", $\text{dir}(e_{t_3}) =$ "d", etc. However, an i-directional path can only use i directions. Thus if ρ is i-directional, necessary $S(\rho) \leqslant i - 1$. □

Combining Fact 4.4, Lemma 4.6, and Fact 4.2 we get the desired result.

Theorem 4.7. *Assume that the network uses the cyclic traffic-light schedule S_3 and that a vertex v sends a message to a vertex w along a route ρ, and $|\rho| = d$. If ρ is a shortest path then the message arrives to w within no more than $2 \cdot d + 3$ time units, and if ρ is three-directional then the message arrives within no more than $2 \cdot d + 4$ time units.*

4.2. Telephone routing on trees

Our main goal in the following two subsections is to construct two instances of the TRS problem on the grid, with routes that are 2- and 3-directional, for which the upper bounds of Theorem 4.7 are the best possible. The construction is presented on two steps. First we study some basic properties of the TRS problem on trees. Then, in Section 4.3 we define the notion of a simple path embedding of a tree in a graph and use such an embedding to build the desired problem instances.

Throughout the rest of this section, $T = (V_1, E_1)$ denotes a rooted tree with root r. We associate with T a TRS problem instance denoted A_T in the following way. For each leaf l in T create a message M_l. The message has to be delivered from l to the root r along the (unique) path between them in the tree. Note that the time needed for completing the schedule in this instance equals the broadcast time from r in the tree

T (see [6]). In order to construct a TRS problem on a tree T that is hard to schedule, we need to enforce some degree constraints on the tree vertices. For this purpose we introduce the following definition.

Definition 4.8 (*Property* $p(k, \mathcal{K})$). Given an integer k, and a set of ordered pairs of integers $\mathcal{K} = \{(s_i, k_i): 1 \leqslant i \leqslant m\}$ such that $1 \leqslant k < k_i$ for every i, the tree satisfies property $p(k, \mathcal{K})$ if the following holds.

(1) For every vertex v in T, the number of children of v in the tree is either k or k_i for some i.

(2) Every path from a leaf $l \in V_1$ to the root r contains at least s_i vertices with k_i children, for every pair (s_i, k_i) in the set $\mathcal{K}$. It is required that the sets of such vertices corresponding to k_i and k_j, $k_i \neq k_j$, are disjoint (that is, the path contains at least s_1 vertices of degree k_1, and at least s_2 *different* vertices of degree k_2, etc.).

Lemma 4.9. *Suppose* $T = (V_1, E_1)$ *is a full tree of height* n (*i.e., all leaves are at depth precisely* n), *and furthermore, it satisfies property* $p(k, \mathcal{K})$ *for* $\mathcal{K} = \{(s_i, k_i): 1 \leqslant i \leqslant l\}$. *Then*

$$T(A_T) \geqslant n \cdot k + \sum_i s_i \cdot (k_i - k).$$

Proof. It suffices to show that the right hand side expression is a lower bound on the broadcast time from the root. Consider an optimal scheme for broadcasting from the root. The proof is established by constructing one "slow" path in this broadcast process. The path is constructed iteratively. At each step, we concatenate to the current end vertex the edge that is scheduled last according to the optimal scheme. This is repeated until a leaf is reached. The time before the message can reach this leaf is bounded by the sum of degrees of the vertices in the chosen path. By the assumed property, this time is bounded by $n \cdot k + \sum_i s_i \cdot (k_i - k)$. □

Let us apply the above lemma to the tree Sp_j defined in Fig. 4. This tree is composed of four trees T_{j-1}, where T_j is defined recursively as depicted in the figure. Here B_j is the full binary tree of height j. Note that Sp_j satisfies property $p(k, \mathcal{K})$, with $k = 2$, and $\mathcal{K} = \{(1, 3), (1, 4)\}$. Thus by Lemma 4.9, $T(A_{\mathrm{Sp}_j}) \geqslant 2 \cdot j + 3$.

We also apply the lemma to the tree $\mathrm{D3}_j$ defined in Fig. 5. This tree satisfies property $p(k, \mathcal{K})$ with $k = 2$, and $\mathcal{K} = \{(2, 3), (1, 4)\}$. Thus by Lemma 4.9, $T(A_{\mathrm{D3}_j}) \geqslant 2 \cdot j + 4$.

4.3. The lower bound

Definition 4.10. A function $F: V_1 \to V$ is a *simple path embedding* of the tree $T = (V_1, E_1)$ in the graph $G = (V, E)$ if the following conditions hold.

(1) $(F(v_1), F(v_2)) \in E$ for every $v_1, v_2 \in V_1$ such that $(v_1, v_2) \in E_1$,

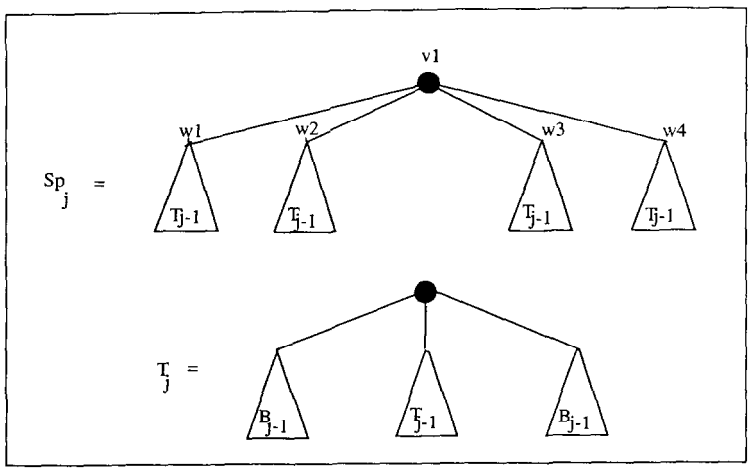

Fig. 4. The trees T_j and Sp_j. B_j is the full binary tree of height j.

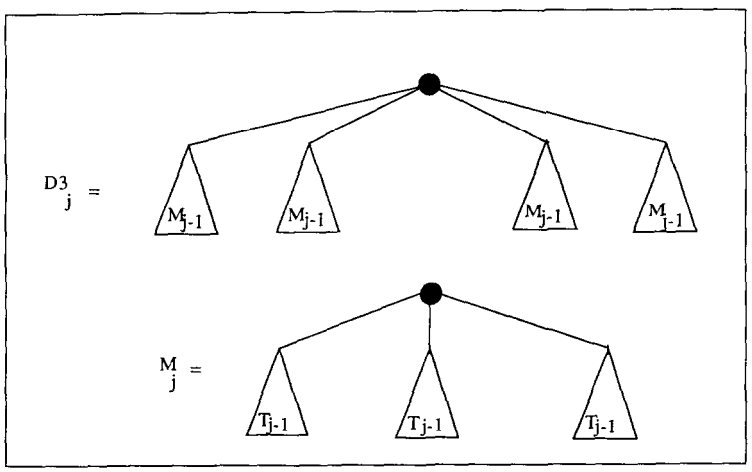

Fig. 5. The trees M_j and $\mathrm{D3}_j$. The tree T_j is defined in Fig. 4.

(2) $F(v_1) \neq F(v_2)$ for every $v_1, v_2 \in V_1$ that are on the same path from a leaf to the root, or share the same parent.

Note that in the above definition, each path from a leaf to the root in T corresponds to a simple path on G. Given a simple path embedding of T in a graph $G = (V, E)$, associate with G a TRS problem instance by assigning a message M_l' to each vertex $F(l) = l' \in V$ such that l is a leaf in T, and requiring that M_l is to be sent from l' to $F(r)$ in G, along the path that corresponds to the path connecting l and r in T. Denote the

associated problem instance by $A_{T,G}$. The following fact easily follows by induction on the height of T.

Fact 4.11. $T(A_{T,G}) \geqslant (TA_T)$.

We construct a simple path embedding F of the tree Sp_j, from Fig. 4, in the grid in the following way. Choose a vertex v in the grid whose properties are to be defined later, and put $F(v_1) = v$. Denoting the neighbors of v in the grid by z_1, z_2, z_3 and z_4, set $F(w_i) = z_i$. For each w_i, we must embed the subtree T_{j-1} rooted at w_i, in the grid. This embedding is performed in a symmetric fashion for $1 \leqslant i \leqslant 4$, and thus we only need to define the embedding for w_1.

Assume without loss of generality that z_1 is v's left neighbor. We define F inductively on the *level* of the vertices (defined as their distance from w_1), in the following way. On level 1, assign the left neighbor of z_1 to the middle child of w_1 (i.e., the child of degree 4), and arbitrarily assign z_1's upper and lower neighbors to the two other children of w_1. For the induction step, assume the embedding is already defined for the first i levels. Consider a vertex u at level i. If $F(u)$ has the same row index as z_1, then u has three children in the tree, and we assign F values to them in the same way we did with w_1's children. Else assume that $F(u)$ is above (resp., below) and to the left of z_1. Arbitrarily assign to the two children of u, $F(u)$'s left and upper (resp., lower) neighbors. Clearly, this is a simple path embedding. We must choose m and n large enough, and choose the vertex v so that this embedding would be possible for the given j. Clearly the embedded paths are shortest paths. Using Fact 4.11 we deduce that $T(A_{\mathrm{SP}_j, \mathrm{Gr}_{m,n}}) \geqslant 2 \cdot j + 3$. Summarizing this discussion and using Theorem 4.7, the following theorem is obtained.

Theorem 4.12. *For sufficiently large m and n, the cyclic traffic-light schedule S_3 is optimal for* TRS(2-DIR) *is the worst case.*

Using similar methods as above it is possible to define a simple path embedding of the tree $\mathrm{D3}_j$ in the grid, where each embedded path in the grid corresponding to a leaf-to-root path in $D3_j$ uses at most three directions. It follows from Fact 4.11 that $T(A_{\mathrm{D3}_j, \mathrm{Gr}_{m,n}}) \geqslant 2 \cdot j + 4$, and we conclude the following.

Theorem 4.13. *For sufficiently large m and n, the cyclic traffic-light schedule S_3 is optimal for* TRS(3-DIR) *in the worst case.*

5. Scheduling arbitrary simple paths on the grid

In this section we deal with arbitrary (4-directional) simple paths on the grid. We first give an upper bound by analyzing the performance of the cyclic traffic-light schedule S_3, and then given a near-matching lower bound.

5.1. The upper bound

In this subsection we analyze the performance of the cyclic traffic-light schedule S_3 on TRS problems where the input routes are arbitrary simple paths. Fact 4.2 suggests that it is needed to bound $\omega(\rho)$ for a given route ρ. As some experimentation hints, $\omega(\rho)$ achieves its maximum value (asymptotically) on a route ρ of the shape depicted in Fig. 6. In order to prove this formally, we show how starting with an arbitrary route ρ, one can perform a series of transformations on the route, each leaving $\omega(\rho)$ unchanged, but bringing ρ to a "simpler" form which in a sense is closer to that of Fig. 6. For this purpose we give some preliminary lemmas and definitions.

Definition 5.1. Given a simple path ρ on a grid, a "North U-turn" N-U-T$(i; j, \ldots, m)$, where $j+1 \leqslant m$, is a subpath $p = (e_1, e_2, \ldots, e_{m-j+2})$ of ρ with the following properties:

(1) edge $e_1 = ((i+1, j), (i, j))$,

(2) for all $2 \leqslant l \leqslant m-j+1$, edge $e_l = ((i, j+l-2), (i, j+l-1))$,

(3) edge $e_{m-j+2} = ((i, m), (i+1, m))$.

The *length of p*, denoted $l(p)$, is $m-j$. "East", "South" and "West" U-turns are defined in an analogous fashion and denoted by E-U-T, S-U-T and W-U-T, respectively (see Fig. 7).

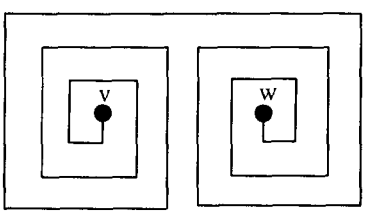

Fig. 6. A route on which ω is maximal.

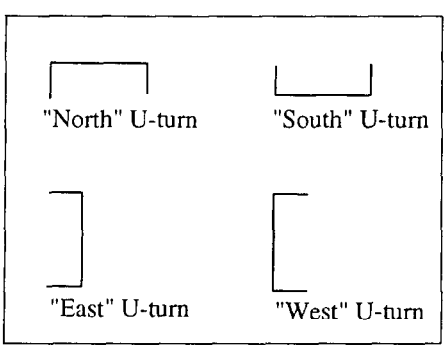

Fig. 7. U-turns on the grid.

Definition 5.2. Let $p = \text{N-U-T}(i_1; j_1, \ldots, m_1)$ and $q = \text{N-U-T}(i_2; j_2, \ldots, m_2)$. Then q is the *parent* of p (and p is a *child* of q) iff they lie on adjacent rows, and p's row segment is fully contained in q's, or more formally:

(1) $i_1 = i_2 + 1$,

(2) $j_2 + 1 \leqslant j_1$,

(3) $m_1 + 1 \leqslant m_2$.

(In fact, requirement (3) is imposed by requirements (1) and (2) plus the fact that the route is simple.) The parenthood relation for S-U-T, W-U-T and E-U-T is defined in an analogous fashion, see Fig. 8.

A U-turn is *external* if it has no parent. The *ancestor* (resp., *descendant*) relation is the transitive closure of the parent (resp., child) relation. The U-turn $p = \text{N-U-T}(i; j, \ldots, m)$ contains the start (resp., end) point of the route if the route starts (resp., ends) at vertex $(i + 1, s)$ for some $j \leqslant s \leqslant m$. Note that parent and child U-turns are required to lie on *adjacent* rows or columns, and hence the U-turns depicted in Fig. 9 are not in parenthood relation, and generally, there may be many "nested" U-turns which are not related by the parenthood relation.

We need the following simple facts.

Fact 5.3. *If p is a parent of q then $l(p) \geqslant l(q) + 2$.*

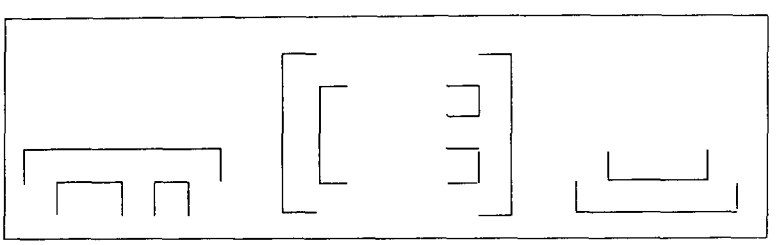

Fig. 8. Examples of the parenthood relation.

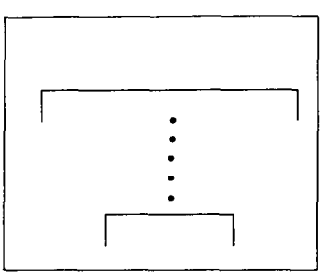

Fig. 9. Nested non parent-child N-U-T.

Fact 5.4. *Let ρ be a path on the grid. Then the parent of each nonexternal U-turn is unique.*

Proof. Two different parents would have to intersect each other. □

Definition 5.5. A simple path ρ in the grid is *canonical* if every U-turn that *does not contain the start or end points* has (at least) one child.

We now define an algorithm for modifying a given simple path ρ on the grid into a canonical path. For that purpose we define a transformation Φ on U-turns. We state the definition for North U-turns; analogous transformations apply to the other directions.

Definition 5.6. Let $p = \text{N-U-T}(i; j, \ldots, m)$, where p does not contain the start or end points, and the path ρ does not lie on the vertices

$$(i+1, j+1), (i+1, j+2), \ldots, (i+1, m-1).$$

Then the transformation Φ applied to p in ρ replaces the subpath p by the subpath

$$(i+1, j), (i+1, j+1), \ldots, (i+1, m).$$

The resulting path is denoted $\Phi(p, \rho)$.

Note that the resulting path $\Phi(p, \rho)$ is kept simple. The resulting three possible situations are depicted in Fig. 10.

Lemma 5.7. *Given a route ρ, the transformation ω preserves $\omega(\Phi(p, \rho)) = \omega(\rho)$.*

Proof. One can directly check that the fact that p does not contain the start or the end point of the route implies that applying the transformation Φ to p cancels out an equal

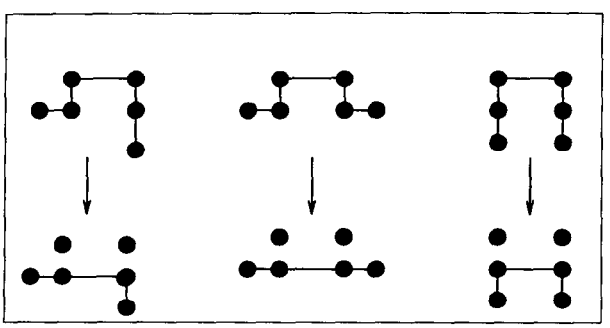

Fig. 10. The transformation Φ on simple paths on the grid.

number of clockwise and counterclockwise turns, specifically, 0, 1 or 2 turns of each type. □

Fact 5.8. *An application of the transformation Φ to ρ decreases its length by* 2.

We now present the modification algorithm. The algorithm takes as input a route ρ and makes several applications of Φ to it.

Algorithm 5.9. *Route canonization algorithm*

(1) $\rho' \leftarrow \rho$

(2) **while** there exists a U-turn p on ρ' to which transformation Φ can be applied **do** $\rho' \leftarrow \Phi(p, \rho')$

(3) return (ρ').

Lemma 5.10. (1) *Applying the canonization Algorithm* 5.9 *to a path ρ leaves $\omega(\rho)$ unchanged,*

(2) *the output ρ' of Algorithm* 5.9 *is canonical.*

Proof. The first claim follows directly from Lemma 5.7.

Let us now prove the second claim. Consider a U-turn p of ρ'. Without loss of generality assume $p = \text{N-U-T}(i; j, \ldots, m)$. Furthermore assume that p does not contain the start or end points of ρ'. Since it is not possible to apply the transformation on p, necessarily $(i+1, s) \in \rho'$ for some $j < s < m$. Choose a minimal such s.

The point $(i+1, s)$ cannot be connected to $(i+1, j)$ (i.e., it cannot be that $s = j+1$ and $((i+1, j), (i+1, j+1)) \in \rho'$), since otherwise ρ' contains a W-U-T on which it is possible to apply the transformation Φ. Since $(i+1, s)$ is not the start (or end) point, $(i+1, s)$ is connected in ρ' to both $(i+2, s)$ and $(i+1, s+1)$. Choose the largest index r such that $(i+1, s)$ is connected to $(i+1, r)$ by a path of the form $(i+1, s)$, $(i+1, s+1), \ldots, (i+1, r)$. The point $(i+1, r)$ must be connected to $(i+2, r)$. Thus p has (at least) one child, namely the U-turn N-U-T$(i+1; s, \ldots, r)$. □

Lemma 5.11. *Given a canonical route ρ having three (different) U-turns of the same type, p, q and r, one of these U-turns is a descendant of one of the other two.*

Proof. Since every U-turn that does not contain an extremal point has a child, every U-turn has a descendent that contains the start or end point. Thus, two of the U-turns, say p and q, have descendants containing the same point. Denote those descendants by p_1 and q_1, respectively. Clearly $p_1 = q_1$, otherwise these U-turn's intersect. For every $i \geq 1$ denote the parent of p_i by p_{i+1} and similarly for q_i. By Fact 5.4, $p_2 = q_2$, $p_3 = q_3, \ldots$ until (without loss of generality) $p_j = q_j = q$ for some $1 \leq j$. Thus q is a descendant of p, completing the proof. □

Lemma 5.12. *Assume that ρ is a canonical path on the grid. Then*

(1) *ρ contains at most two external N-U-T's,*

(2) *if ρ contains two external N-U-T's, then every N-U-T of ρ has at most one child,*

(3) *if ρ contains only one external N-U-T then there is at most one N-U-T having two children, and all the other N-U-T's have at most one child,*

(4) *a similar claim applies to E-U-T's, W-U-T's and S-U-T's.*

Proof. The route ρ cannot have 3 external N-U-T's because of Lemma 5.11. If ρ contains two external N-U-T's then again, assuming that a N-U-T has two children would contradict Lemma 5.11. The third part of the lemma follows analogously. □

For deriving our final bound, we will need the following technical fact obtained by a direct application of the Cauchy–Schwartz inequality (cf. [4]) to $(1, \ldots, 1)$ and $(x_1, \ldots, x_n)$.

Fact 5.13. *For every* $x_1, x_2, \ldots, x_n \in R$, $(x_1 + x_2 + \cdots + x_n)^2 \leqslant n \cdot (x_1^2 + x_2^2 + \cdots + x_n^2)$.

Lemma 5.14. *For each route ρ such that $|\rho| = d$, $\omega(\rho) \leqslant 2 \cdot \sqrt{2} \cdot \sqrt{d}$.*

Proof. By Lemma 5.10 and Fact 5.8, it suffices to prove the claim for canonical paths. Suppose that ρ is canonical. Recall that in using the cyclic traffic-light schedule S_3, it takes three time units to complete a clockwise turn, and only one time unit to complete a counterclockwise turn. Let us denote a clockwise turn by "G" and counterclockwise turn by "B". We are interested in the sequence of G's and B's defined by the route.

Consider the following accounting process. We start traversing the route keeping a counter L serving as a lower bound for d and a counter C serving as an upper bound on $\omega(\rho)$. Set $C \leftarrow 1$ and $L \leftarrow 0$. Each time we encounter a "G" turn, such that the preceding turn was a "B" turn (or vice versa), the two turns cancel out, and we do not increment C or L. Else if we encounter a "G" (resp., "B") turn such that the preceding turn was also a "G" ("B") turn, indicating that we have encountered a U-turn p, we increment L by $l(p)$ and C by one. It is easy to prove by induction on the number of incrementation steps of C that for each route ρ ending in a "B" turn, $C(\rho) \geqslant \omega(\rho) + 1$, and for an arbitrary ρ, $C(\rho) \geqslant \omega(\rho)$.

We get all the possible U-turns by taking all the external U-turns and all their respective descendants. We would now like to estimate the increase in C relative to the increase in L. This is done by considering each of the directions separately. Let us first consider the north direction.

Assume that ρ contains only one external N-U-T p. By Lemma 5.14, only one of its descendants may have two children, while all other can have at most one child.

Assume that $p_2, p_3, \ldots, p_i$ are the first $i-1$ descendants of p and that p_i has two children c_1 and d_1 each having descendants $c_2, c_3, \ldots, c_j$ and $d_2, d_3, \ldots, d_k$ respectively. Thus c_j and d_k each contain the start or end points. In this case, our accounting process has incremented L by

$$(l(p_1) + \cdots + l(p_i)) + (l(c_1) + \cdots + l(c_j)) + (l(d_1) + \cdots + l(d_k)).$$

Note that $l(c_j), l(d_k) \geqslant 1$, and from Fact 5.3, $l(p_i) \geqslant 2 \cdot j + 2 \cdot k + 1$. The contribution of the c turns to the increase in L is at least $((1 + 2 \cdot j - 1) \cdot j)/2 = j^2$. Similarly L is incremented by k^2 due to the d turns. Finally note that L is incremented by at least $(2 \cdot (2 \cdot j + 2 \cdot k + 1) + 2 \cdot (i - 1)) \cdot i/2 = i^2 + 2 \cdot j \cdot i + 2 \cdot k \cdot i$ due to the p turns. In total we increment L by at least

$$i^2 + j^2 + k^2 + 2 \cdot j \cdot i + 2 \cdot k \cdot i.$$

Denote $i + j = h$. It follows that L is incremented by more than $h^2 + k^2$. Note that the counter C is incremented by exactly $i + j + k = h + k$ due to north U-turns.

Let us now consider the case where there are two external N-U-T's. In this case each N-U-T has at most one child, and thus by similar methods it follows that there exist two integers h and k such that $h^2 + k^2$ is a lower bound on the increase in L due to N-U-T's, and $h + k$ is the increase in C due to these turns.

Applying a similar analysis to the other three directions, it follows that there exist up to 8 integers m_i, $1 \leqslant i \leqslant 8$, such that L is incremented by at least $\sum_i m_i^2$, and $C = \sum_i m_i$. Thus by Fact 5.13,

$$\omega(\rho)^2 \leqslant \left(\sum_i m_i\right)^2 \leqslant 8 \cdot \sum_i m_i^2 \leqslant 8 \cdot d$$

and the proof follows. □

By Corollary 4.2 we now have

Corollary 5.15. *Assume that the grid network employs the cyclic schedule S_3. Let v and w be a pair of vertices on a grid. Suppose that v sends a message to w along an arbitrary simple path ρ. Then the message arrives after no more than $2 \cdot |\rho| + 2 \cdot \sqrt{2} \cdot \sqrt{|\rho|} + 2$ time units.*

We comment that for every constant $c < 2 \cdot \sqrt{2}$, there exists a route ρ between two vertices v and w such that the cyclic traffic-light schedule S_3 completes the routing along the path in more than $2 \cdot |\rho| + c \cdot \sqrt{|\rho|}$ time units. To see this, denote the route in Fig. 6 by ρ. Observe that if the number of South U-turns in ρ is $2 \cdot j$ then $|\rho| = 8 \cdot j^2 + 4 \cdot j - 1$ and $\omega(\rho) = 8 \cdot j - 2$. The comment follows by Corollary 4.2 for sufficiently large j.

5.2. The lower bound

The purpose of this subsection is to construct an example of a TRS problem involving simple paths on the grid, for which the best possible schedule time approaches the upper bound presented in the previous subsection. For this purpose we introduce the definition of a *bounding frame* of a simple path on a grid.

Definition 5.16. The *frame* of a simple path on a grid is defined as follows.

(1) If the path is empty then the frame is empty too.

(2) Else take a leftmost vertex $v_1 = (w_1, w_2)$, a rightmost vertex $v_2 = (w_3, w_4)$, a highest vertex $v_3 = (w_5, w_6)$, and a lowest vertex $v_4 = (w_7, w_8)$ on ρ. The frame is the rectangle with corners (w_7, w_2), (w_5, w_2), (w_7, w_4) and (w_5, w_4). (See Fig. 11.)

The *length* of the frame is $w_4 - w_2$ and its *height* is $w_5 - w_7$. A vertex $v = (v_1, v_2)$ is a *border point* of the frame if $v_1 = w_5$ or $v_1 = w_7$ or $v_2 = w_2$ or $v_2 = w_4$. All nonborder points contained in the frame are referred to as *internal* points of the frame, while points that are outside the frame are called *external*. Denote the border of a route ρ by $f(\rho)$.

Let d be a positive integer. We want to construct a tree of height d and embed it in the grid, in such a way that the TRS instance resulting from all leaf-to-root paths in the tree is "hard" to schedule. We choose a grid $\mathrm{Gr}_{m,n}$ with sufficiently large m and n, and a vertex v in the grid such that the following construction is possible. Let us define a sequence of d sets S_i, $1 \leqslant i \leqslant d$, of paths all starting at v. We define the paths in S_i inductively with the paths in S_{i+1} being the paths of S_i, each extended in several ways by additional edges. All the paths in S_i have length i. We also maintain the following *out-propagation* property:

For every path ρ in S_i, end vertex w of ρ is on $f(\rho)$.

Algorithm 5.17. *Embedded tree construction*

(1) $S_0 \leftarrow \emptyset$.

(2) *Stage* 1. Create S_1 by adding to S_0 the four edges connecting v to its four neighbors in the grid (the edges are regarded as paths of length one).

(3) *Stage* $i + 1$. Assume that the sets S_k, for $1 \leqslant k \leqslant i < d$, are already defined. For every path $\rho = (e_1, \ldots, e_i)$ in S_i perform the following. Let $w = (w_1, w_2)$ denote

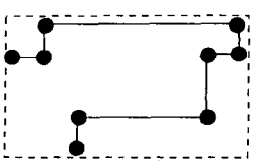

Fig. 11. Frame of a simple path.

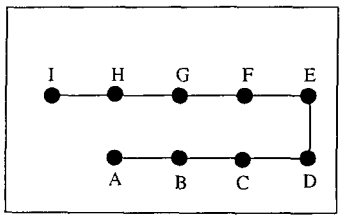

Fig. 12. A path in S_8. The subpaths ending in nodes B, C, D, E, I, were extended 3 times. The subpaths ending at nodes F, G, H were extended twice. The initial empty path was extended from A in 4 ways.

the end vertex of the path ρ. Extend this path to a number of new paths, by connecting w to some of its neighbors, and add the augmented paths to S_{i+1}.

In doing so, it must be verified that the resulting paths remain simple. Thus add to S_{i+1} the path obtained by extending ρ by connecting w to his left neighbor iff ρ does not contain a point (w_1, w_3) for some integer $w_3 < w_2$, i.e., extend ρ leftward iff there is no vertex in ρ at the same row and to the left of w. A similar action is performed for all the other directions.

If the above method can be used in order to connect w to l of its neighbors for some $0 \leqslant l \leqslant 3$, we say that ρ *can be extended in* l *ways.*

Lemma 5.18. *Let* $w = (i_1, j_1)$ *be the end vertex of a route* $\rho \in S_k$. *Then*

(1) w *is on the border of* ρ, *and*

(2) ρ *can be extended in at least two ways.*

Proof. We prove both claims simultaneously by induction on the length $|\rho|$ of the path. The cases $|\rho| = 0, 1$ are obvious. Assume that the claim holds for any path ρ such that $|\rho| = l$ for $d > l \geqslant 1$. Further assume, e.g., that the end vertex w is on the *right* border of $f(\rho)$. By the induction hypothesis, ρ can be extended in at least two different ways.

If in the next move ρ is extended upwards or to the right, the resulting new end vertex is still on the right border, and furthermore, by definition it is possible to extend the new resulting path upwards and to the right. Thus both claims still hold.

Assume now that ρ is extended from w to the left. Then the last move to w was of either of the types "u" or "d". Assume, without loss of generality, that it was a "u" move. In this scenario we claim that w is at the top border. If this is not the case, then there is a vertex $v = (m', n')$ in ρ s.t. $m' < i_1$ and $n' < j_1$. However, the route connecting v to $(i_1 + 1, j_1)$ must cross the i_1th row or the j_1th column. This leads to contradiction. It follows that the end vertex of the resulting path is on the top border, and furthermore, the resulting path can be extended upwards, and to the left. This completes the inductive proof. □

Fact 5.19. *All the paths is S_d are simple paths of length d.*

In the above construction it is possible to associate with S_d a tree F_d in the following way. Define a vertex v_1 to be associated with v. Assume now that for a given path ρ in S_i we extend the end vertex w of ρ in $2 \leqslant k \leqslant 4$ different ways. Then add to the tree k new (distinct) vertices and connect them to the vertex in the tree, associated with w. Clearly this is a simple path embedding of F_d in the grid.

Lemma 5.20. *The tree F_d satisfies property $p(k, \mathcal{K})$ with $k = 2$ and $\mathcal{K} = \{(2 \cdot \sqrt{d+1} - 3, 3), (1, 4)\}$.*

Proof. Note that whenever a path ρ is extended to a new path ρ' such that $f(\rho') \neq f(\rho)$, in the next step ρ' can be extended in 3 ways. Let $\rho \in S_d$ be a path after the final stage, whose frame $f(\rho)$ is of height l and length k. Since all the points of ρ are within the frame and ρ is a simple path, $(l+1) \cdot (k+1) \geqslant d+1$. Along the path ρ there are at least $l + k$ vertices that were extended in at least 3 different ways (at the times after the frame is extended and on the first stage). Clearly $l + k + 2 \geqslant 2 \cdot \sqrt{d+1}$. Since in the first stage the empty path is extend in 4 ways, the proof follows. □

By Lemma 4.9 and Fact 4.11 we have

Corollary 5.21. $T(A_{F_d, \mathrm{Gr}_{m,n}}) \geqslant 2 \cdot d + 2 \cdot \sqrt{d+1} - 1$.

We can summarize Corollary 5.21 and Corollary 5.15 in the following theorem.

Theorem 5.22. *Routing an instance of* TRS_d(4-DIR) *many require* $2 \cdot d + \mathrm{O}(\sqrt{d})$ *rounds in the worst case.*

Acknowledgement

We are grateful to the anonymous referee for his suggestions and remarks, that have helped us to considerably improve the presentation, and simplify the proof of Lemma 4.9.

References

[1] A.M. Farley and A. Proskurowski, Gossiping in grid graphs, J. Combin. Inform. System Sci. 15 (1980) 161–162.

[2] S. Hedetniemi, S. Hedetniemi and A. Liestman, A survey of gossiping and broadcasting in communication networks, Networks 18 (1988) 319–349.

[3] G. Kortsarz, Telephone routing, M.Sc. Thesis (1990).
[4] D. Kreider, An Introduction to Linear Analysis (Addison-Wesley, Reading, MA, 1966).
[5] A.L. Liestman and D. Richards, Network communication in edge-colored graphs, Unpublished manuscript (1991).
[6] P.J. Slater, E.J. Cockayne and T. Hedetniemi, Information dissemination in trees, SIAM J. Comput. 10 (1981).