

Antonín Kavička - Ludmila Jánošíková \*

# MODELovanie KOĽAJISKA A VÝPOČET NAJKRATŠej JAZDNEJ CESTY

## TRACKAGE MODELLING AND ALGORITHMS FOR FINDING THE SHORTEST TRAIN ROUTE

Článok popisuje dva modely koľajovej siete v železničnej stanici a na nich založené algoritmy výpočtu najkratšej jazdnej cesty. Prvý model - ohodnotený digraf - reprezentuje každú koľaj a výhybkovú koľaj pomocou dvoch vrcholov. Hrany predstavujú prípusné prechody medzi koľajami. Druhý model - hranovo ohodnotený neorientovaný graf - modeluje koľaje a výhybkové koľaje pomocou hrán. Vrcholy v ňom predstavujú spojenia medzi koľajami. Pre obidva modely boli navrhnuté a overené algoritmy hľadania jazdnej cesty, ktoré rešpektujú pravidlá pre pohyb koľajových vozidiel, ako aj dĺžku vlaku a obsadenie koľají inými koľajovými vozidlami.

### 1. Úvod

Jedným z problémov riešených na Katedre dopravných sietí FRI ŽU je počítačová simulácia prevádzky na železnici [1, 2, 4, 5, 6, 7]. Podarilo sa nám vytvoriť a v praxi overiť simulačný model zriaďovacej stanice.

Základnou úlohou zriaďovacej stanice je prijmanie, rozpúšťanie a tvorba nákladných vlakov, ako aj obsluha vlakov tranzitných a skupinových. Vozne, ktoré prišli do stanice vo vlakoch zo zaústujúcich tratí alebo zo záťaze z vlastného uzla, sa triedia podľa miesta alebo podľa smeru určenia a vytvárajú sa z nich nové nákladné vlaky, odvážajúce záťaž do miesta určenia alebo do inej vlakotvornej stanice. Doba pobytu vozňa v zriaďovacích staniciach tvorí významnú časť doby obehu vozňa v rámci železničnej siete. Aby služby poskytované železničnou nákladnou dopravou boli čo najkvalitnejšie, snažíme sa dobu obehu vozňa minimalizovať. Jednou z cest, ako to dosiahnuť, je znížiť počet staníc, v ktorých je vozeň prepravovaný na jeho ceste z východiskovej do cieľovej stanice. Je teda nutné koncentrovať vlakotvorbu do veľkých triediacich stanic a menšie stanice rušiť. Vo veľkých stanicach však potom stúpajú požiadavky na dokonalú organizáciu práce a efektívne využitie všetkých zdrojov. Nás simulačný model poskytuje informácie o kapacitných možnostiach triediacich stanic, o dôsledkoch ich prípadnej rekonštrukcie a o využití technických a ľudských zdrojov. Perspektívne by mohol tiež slúžiť ako podporný prostriedok pre rozhodovanie dispečera v reálnom čase.

This paper deals with two models of track network in a railway station and algorithms for finding the shortest train route. In the first model - a weighted digraph - two vertices represent a track or a switch track. Edges of a digraph are understood as the possible transitions between tracks. In the second model - a non-oriented edge weighted graph - the edges represent tracks and the vertices match the connections between tracks. Algorithms for finding a shortest route were designed and verified for both models. These algorithms respect rules for the movement of rail vehicles, as well as their length and the occupation of tracks by other rolling stock.

### 1. Introduction

The Department of Transport Networks (University of Žilina) has been working in the field of computer simulation applied to the operation of railway systems for many years [1, 2, 4, 5, 6, 7]. We have been developing a complex simulation model of a marshalling yard operation in recent years. The mentioned model has been verified and already applied within the frame of several real projects.

The basic task of a marshalling yard is to receive, sort and form freight trains, as well as to attend transit trains and group trains. The train cars coming from railway tracks, which flow into or originate from the station, are sorted according to the place or the direction of destination. New goods trains transport the freight to the station of destination or to an intermediate station. The sojourn time of a train car in a station represents an important part of the circulation time of a train car on the railway network. To improve the services of railway freight traffic, the circulation time of a train car should be as short as possible. One of the possible solutions of this task is to reduce the number of stations in which the train car is handled on its way from the station of origin to the station of destination. Trains should be formed in big marshalling yards and small marshalling yards should be cancelled. However, the requirements for perfect organisation of operation in big stations then increase. Our simulation model gives information about the utilisation of service means of the marshalling yard

\* Ing. Antonín Kavička, PhD., Ing. Ludmila Jánošíková, CSc.

University of Žilina, Faculty of Management Science and Informatics, Department of Transport Networks

Ďalšou oblasťou, v ktorej možno využiť počítačový simulačný model, je laboratórna výučba študentov - budúcich dispečerov.

Pri tvorbe počítačového simulačného modelu (ďalej len simulačného modelu) sa uplatňujú poznatky z rôznych vedných odvetví, ako sú počítačová simulácia, teória pravdepodobnosti, teória hromadnej obsluhy, teória databáz, programovacie techniky. Tieto vedné odbory majú v simulácii všeobecné uplatnenie. Okrem toho sú však pri tvorbe modelu nevyhnutné vedomosti zo špecifických oblastí súvisiacich so simulovaným procesom. Pri simulácii prevádzky železničnej stanice potrebujeme nástroje a metódy matematickej teórie dopravy a teórie grafov. Avšak často prístupy a metódy z práve menovaných oblastí nemožno v simulačnom modeli aplikovať priamo, ale je potrebné ich upraviť podľa potrieb simulačného modelu.

Otázka vhodného modelovania železničnej koľajovej siete v stanici je jednou z prvých, na ktoré musíme pri tvorbe simulačného modelu stanice odpovedať. Železničnú koľajovú sieť v stanici (koľajisko) chápeme ako dopravnú sieť, po ktorej sa pohybujú koľajové vozidlá. Pohyb koľajových vozidiel v kolajisku je limitovaný jednak istými technickými obmedzeniami (napr. vlaky sa na koľaji nemôžu predbiehať, ďalej nie je možné, aby koľajové vozidlo, ktoré príde na výhybku, z nej ďalej pokračovalo v pohybe po ľubovoľnej koľaji, ktorá je s ňou spojená a pod.), a jednak železničnými technologickými predpismi (napr. pri premestňovaní posunovacej lokomotivy v vchodej skupiny koľají na odchodovú skupinu nemôže lokomotíva prechádzať po koľaji na zvážnom pahorku).

Pri modelovaní koľajiska (podobne ako pri modelovaní iných dopravných sieti) je vhodné použiť nástroje teórie grafov. Pri návrhu modelu koľajiska sa navýše musí zohľadniť:

- špecifický charakter železničnej koľajovej siete (s jej vyššie uvedenými technickými a technologickými obmedzeniami),
- miera podrobnosti, s ktorou chceme sledovať infraštruktúru koľajiska,
- špecifický charakter pohybu koľajových vozidiel v koľajisku a
- požadovaná miera presnosti, s ktorou chceme pohyby koľajových vozidiel sledovať.

Ked' modelujeme pohyb koľajových vozidiel, nemôžeme zanechať ich dĺžku, tzn. nemôžeme ich považovať za hmotné body o nulových dĺžkach. Ďalej si musíme uvedomiť, že presun koľajového vozidla z jeho východiskovej do danej cieľovej pozície nemožno vždy uskutočniť bez zmeny smeru pohybu (niekedy je nutné pohyb v jednom smere zastaviť a potom nechať koľajové vozidlo pokračovať v jeho pohybe opačným smerom než pred zastavením).

Tvorca modelu koľajiska stojí teda pred problémom, aký typ grafu a akú jeho implementáciu na počítači zvoliť, aby boli zohľadnené vyššie uvedené skutočnosti. Pri návrhu modelu musíme, samozrejme, vziať do úvahy aj to, aké úlohy chceme riešiť. Pri riadení prevádzky v železničnej stanici je jednou z najčastejšie riešených úloh vyhľadanie prípustnej, a pokiaľ možno najkratšej trasy pre premestnenie vozidiel v koľajisku. V simulačnom modeli je nevyhnutné automatické riešenie tejto úlohy počítačom.

(tracks, shunting locomotives, working gangs) as well as information about the consequences of changes in trackage infrastructure, work schedule and technological procedures. The model could also serve as a support tool for dispatcher's decisions or as a training tool for future student dispatchers.

In creating a computer simulation model, knowledge of various branches such as computer simulation, probability theory, queuing theory, database theory and programming techniques is valuable. Simulation uses these branches of science. In addition, other specific knowledge connected with the simulated processes is needed for the process of model creation. For simulation of the operation of a marshalling yard, we need the tools and methods of mathematical transport theory and graph theory. However, the approaches and the methods of the mentioned branches often cannot be applied to a simulation model directly. They have to be settled according to needs of a model.

One of the first questions we are supposed to answer when simulating a marshalling yard operation is how to model track network (trackage) in a station. The track network in a station is a transport network where the rolling stock move. The rail vehicles movements are limited by certain technical restrictions (e.g. trains cannot outrun each other on the same track, a rail vehicle that comes to a switch cannot go on using an arbitrary track connected with a switch, etc.). Further, the movement has to respect railway technological rules (e.g. a shunting locomotive cannot pass a hump track transferring from reception tracks to departure tracks).

To model a trackage (like other transport networks) we can use the tools of graph theory. In addition, we have to take into account:

- The specific character of the railway track network (with its above mentioned technical and technological restrictions),
- The level of details which we want to investigate on a trackage infrastructure,
- The specific character of the movement of rail vehicles on a trackage, and
- The required precise degree of rail vehicles movements.

When modelling the movement of rail vehicles we cannot neglect their length. They cannot be regarded as objects with zero lengths. Further, we should realise that sometimes a rail vehicle cannot transfer from its original position to the certain destination position without changing the direction of its movement. Sometimes it is necessary to stop the movement in one direction and then to let the rail vehicle go on in the opposite direction.

The designer has to choose such a graph and its computer implementation in order to take this into consideration. Model design must be adjusted to the problems which need to be solved. One of the most often solved problems (managing a marshalling yard operation) is to find the shortest admissible route for a transfer of the rail vehicle on a trackage. In a computer simulation model, it is necessary to solve this problem automatically with a convenient algorithm.

S ohľadom na špecifický charakter železničnej koľajovej siete, špecifický charakter pohybu koľajových vozidiel a s cieľom poskytnúť podporu pre automatický výpočet najkratšej jazdnej cesty v koľajisku sme navrhli dva rôzne spôsoby modelovania koľajiska. Prvým z nich je ohodnotený digraf (orientovaný graf), v ktorom je každá koľaj reprezentovaná pomocou dvoch vrcholov a hrany predstavujú pripustné prechody medzi koľajami. Druhý model - hranovo ohodnotený neorientovaný graf - modeluje koľaje pomocou hrán. Vrcholy v ňom predstavujú spojenia medzi koľajami. Pre každý model sme vypracovali algoritmus pre výpočet najkratšej jazdnej cesty. Obidva modely a princípy algoritmov sú ukázané v nasledujúcich kapitolách.

## 2. Ohodnotený digraf ako model koľajiska

Ohodnotený digraf predstavuje dosť zložitý model koľajiska. Výhodou však je, že optimálnou trasou na premiestnenie vlaku z jednej koľaje na druhú je v tomto modeli najkratšia cesta z jedného vrcholu do druhého, ktorú možno vyhľadať pomocou známych metód, napr. Dijkstrovým algoritmom.

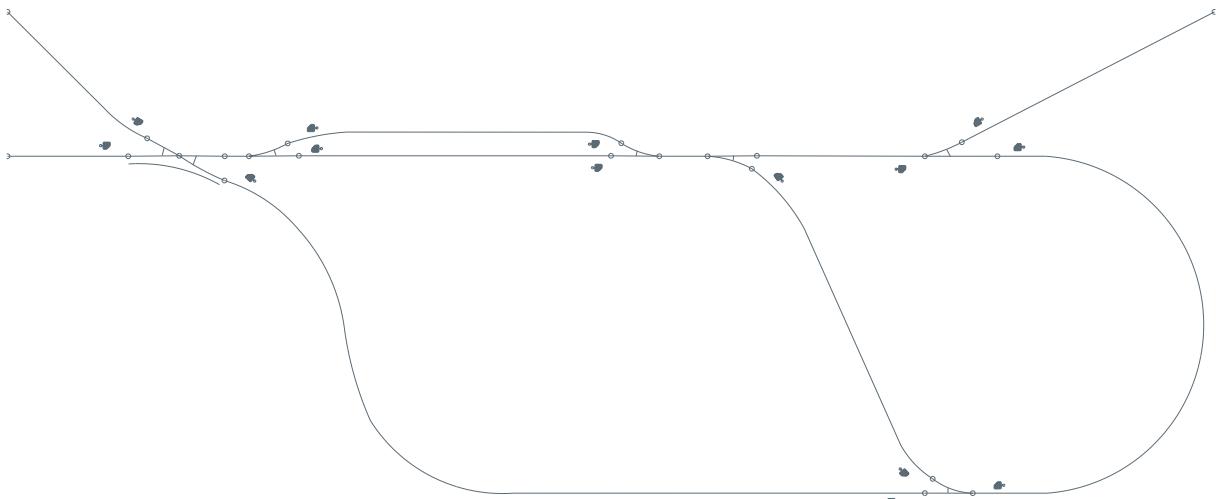
Model budeme ilustrovať na časti koľajiska podľa obr. 2.1. Plán na obr. 2.1 bol zostrojený pomocou grafického editora AutoCAD LT. Sú v ňom znázornené koľaje, výhybky a návestidlá.

With regard to the specific character of a railway track network, the specific character of the movement of rail vehicles, and the aim of giving support for automatic calculation of the shortest route on a trackage, two different methods of trackage modelling were designed. The first model is represented by a weighted digraph. Two vertices of a digraph represent a track. The edges reflect the possible transitions between the tracks. The second model is a non-oriented edge weighted graph whose edges represent tracks and vertices match the connections between tracks. For both models, algorithms for finding the shortest route were designed. The mentioned models and the principles of algorithms are presented in the following chapters.

## 2. A weighted digraph as a trackage model

A weighted digraph is a rather complex model of a trackage. However, this model also presents an advantage. The shortest path between a pair of vertices represents an optimal route for train transfer from one track to another. The shortest path can be calculated by well-known methods (e.g. Dijkstra's algorithm).

The model will be illustrated on a part of a trackage shown in Fig. 2.1. The plan in Fig. 2.1 was drawn using a graphic editor AutoCAD LT. The tracks, switches and signals can be seen on the plan.



Obr. 2.1. Časť koľajiska  
Fig. 2.1 A part of a trackage

Pri zostavovaní modelu sa uplatnia niektoré myšlienky Zelinku, ktorý sa zaoberal polárnymi grafmi [7]. Postup návrhu modelu môžeme zhrnúť do štyroch krokov:

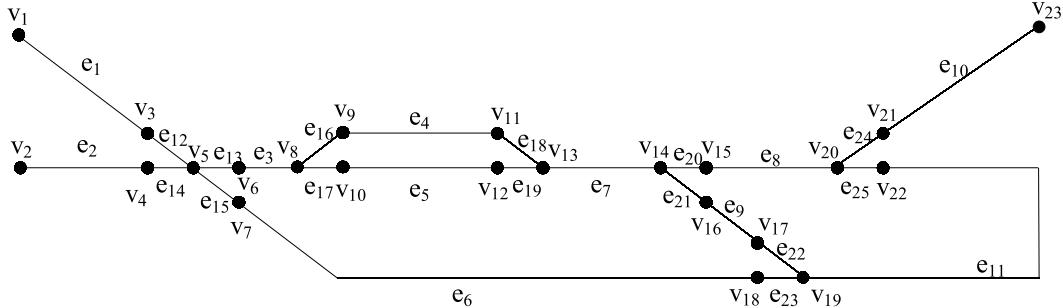
1. Vytvoríme graf  $G_0$ , v ktorom hrany reprezentujú koľaje a vrcholy predstavujú spojenia medzi koľajami (obr. 2.2). Koľaje, ktoré tvoria výhybku, sú reprezentované samostatnými hranami. V tejto fáze návrhu ešte nie sú definované pravidlá pre prechody cez výhybky.

The ideas of B. Zelinka are used [8] concerning polar graphs within the frame of model design process. That process can be summarised into four steps:

1. Construct graph  $G_0$  whose edges represent the tracks and the vertices represent the connections between tracks (Fig. 2.2). The switch tracks are represented by independent edges. In this step, the rules for passing through switches have not yet

vidlá pre prejazd cez výhybky. Najkratšou trasou pre pre-miestnenie vlaku je v grafe  $G_0$  sled medzi dvoma hranami.

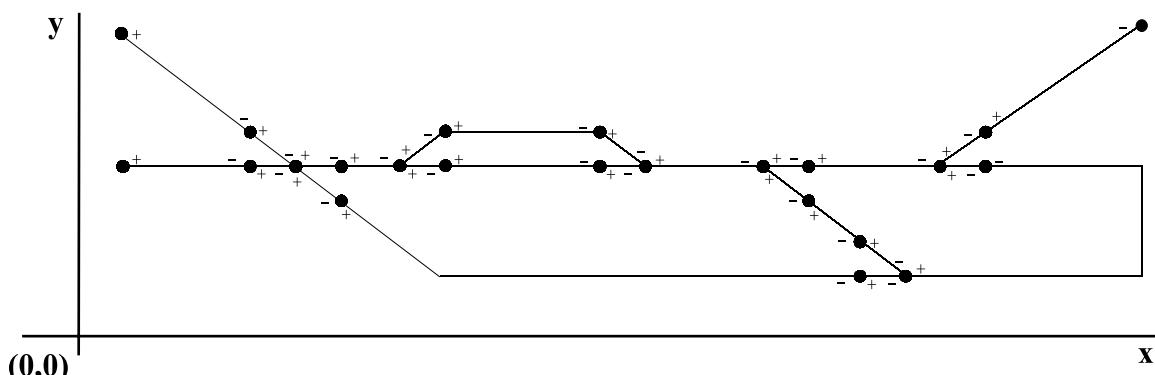
been defined. The shortest walk between the pair of edges in graph  $G_0$  represents the shortest route for train transfer.



Obr. 2.2. Graf  $G_0$   
Fig. 2.2 Graph  $G_0$

2. Opačné konce hrán grafu  $G_0$  označíme opačnými znamienkami. Pomôžeme si pritom tak, že si predstavíme umiestnenie grafu  $G_0$  v súradnicovom systéme (obr. 2.3). Každý vrchol grafu potom môžeme stotožniť s koncami hrán, ktoré sú s ním incidentné, napr.  $v_3 \equiv e_1^- \equiv e_{12}^+$ .

2. Label the opposite endpoints of each edge of graph  $G_0$  with opposite signs. We can imagine that graph  $G_0$  is placed in a two-dimensional co-ordinate system (Fig. 2.3). The edge endpoint with a lesser x-coordinate gets the sign (+) and the opposite one (-). Each vertex can then be identified with the endpoints of incident edges, e.g.  $v_3 \equiv e_1^- \equiv e_{12}^+$ .



Obr. 2.3. Graf  $G_0$  - označenie opačných koncov hrán  
Fig. 2.3 Graph  $G_0$  - labelling of the opposite endpoints of edges

3. Vykonáme transformáciu grafu  $G_0$  na polárny graf  $G_1$  (obr. 2.4). Každej hrane grafu  $G_0$  odpovedá v grafe  $G_1$  vrchol s dvoma pólmi. Póly vrcholu reprezentujú opačné konce hrany. V grafe  $G_1$  vrcholy reprezentujú koľaje a výhybkové koľaje. Dva vrcholy grafu  $G_1$  sú spojené hranou vtedy, keď sú odpovedajúce hrany v grafe  $G_0$  susedné. Napr. v grafe  $G_0$  sú susedné hrany  $e_1$  a  $e_{12}$ , preto v grafe  $G_1$  bude hrana  $[v_1^-, v_{12}^+]$ , ktorá je incidentná s pólom - vrcholu  $v_1$  a s pólom + vrcholu  $v_{12}$ . Pri určovaní pólov krajných vrcholov v hrany v grafe  $G_1$  využijeme označenie vrcholov grafu  $G_0$  pomocou koncov hrán, ako bolo naznačené v predchádzajúcom odseku ( $v_3 \equiv e_1^- \equiv e_{12}^+$ ). Hrana v grafe existuje vtedy, keď je technicky prípustné, aby koľajové vozidlo prešlo priamo z koľaje mode-

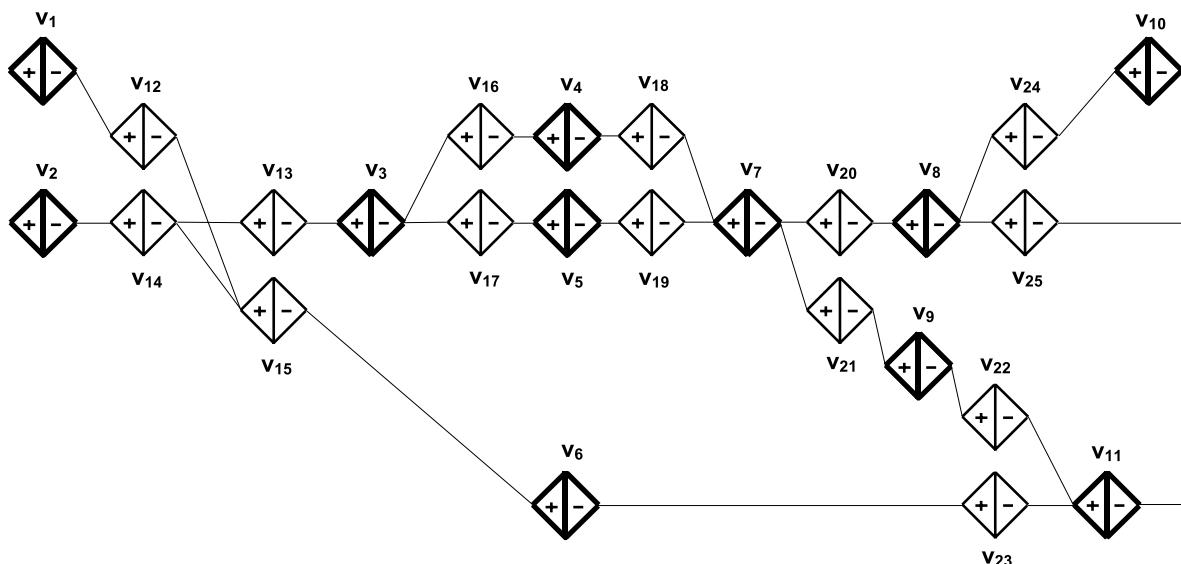
3. Transform graph  $G_0$  into the polar graph  $G_1$  (Fig. 2.4). Each edge of graph  $G_0$  corresponds to a vertex with two poles in graph  $G_1$ . Vertex poles represent opposite endpoints of the edge. The vertices of graph  $G_1$  represent tracks and switch tracks. Two vertices of  $G_1$  are connected with an edge if the corresponding edges of  $G_0$  are adjacent. For example, the edges  $e_1$  and  $e_{12}$  are adjacent in  $G_0$  thus the edge  $[v_1^-, v_{12}^+]$  that is incident with the - pole of vertex  $v_1$  and the + pole of vertex  $v_{12}$  will be in  $G_1$ . In addition, an edge exists in  $G_1$  only if it is technically allowed for a rail vehicle to transit from the track modelled by the first vertex to the track modelled by the second one. For example, on the single-slip switch, which is composed of tracks  $e_{12}$ ,  $e_{13}$ ,  $e_{14}$  and  $e_{15}$ , a rail vehicle can go

lovanéj prvým vrcholom na kolaj modelovanú druhým vrcholom. Napr. na polovičnej anglickej výhybke tvorenej kolajmi  $e_{12}$ ,  $e_{13}$ ,  $e_{14}$  a  $e_{15}$  môžeme prejsť z kolaje  $e_{12}$  len na kolaj  $e_{15}$ . Pretože typ výhybky neumožňuje prechod z  $e_{12}$  na  $e_{13}$  ani z  $e_{12}$  na  $e_{14}$ , nie je v grafe  $G_1$  hrana medzi vrcholmi  $v_{12}$  a  $v_{13}$  ani medzi  $v_{12}$  a  $v_{14}$ , ale len hrana  $[v_{12}^-, v_{15}^+]$ .

4. Výsledným modelom je digraf  $G_2$  (obr. 2.5), ktorý dostaneme transformáciou polárneho grafu  $G_1$ .

from track  $e_{12}$  just to track  $e_{15}$ . It means that we can construct the edge  $[v_{12}^-, v_{15}^+]$ . It is not allowed on this type of switch to transit either from  $e_{12}$  to  $e_{13}$  or from  $e_{12}$  to  $e_{14}$ . Therefore, the edges between vertices  $v_{12}$ ,  $v_{13}$  and between  $v_{12}$ ,  $v_{14}$  do not exist.

4. Transform polar graph  $G_1$  to digraph  $G_2$  (Fig. 2.5), which represents the final model.



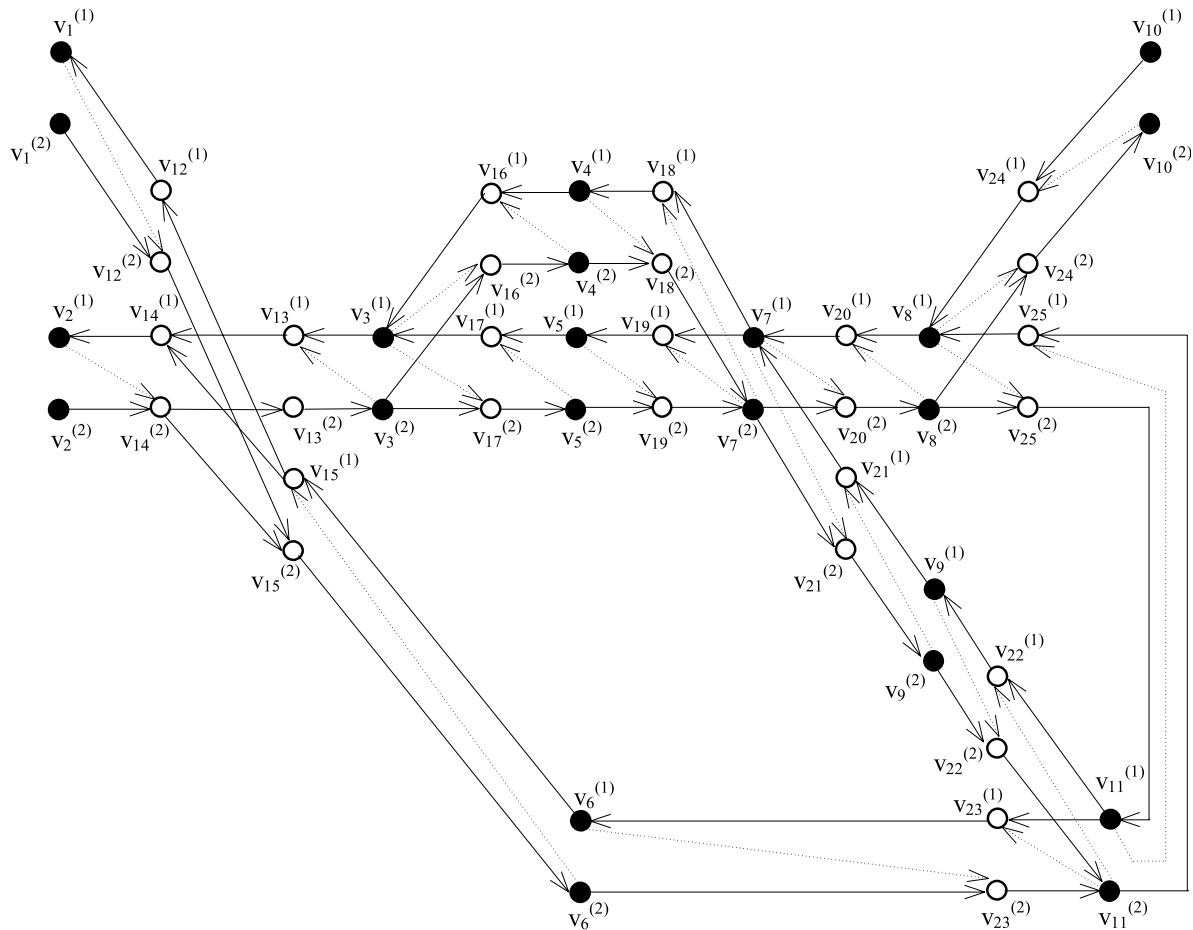
Obr. 2.4. Polárny graf  $G_1$   
Fig. 2.4 Polar graph  $G_1$

Každý vrchol  $v_i$  grafu  $G_1$  je v grafe  $G_2$  reprezentovaný dvojicou vrcholov  $v_i^{(1)}$ ,  $v_i^{(2)}$ , čo znamená, že kolaj možno prechádzať dvoma opačnými smermi. Hrany grafu  $G_2$  môžeme rozdeliť na *tranzitné* (znázornené plnou čiarou) a *reverzné* (znázornené čiarovanou čiarou). Tranzitné hrany vyjadrujú možnosť prejazdu (tranzitu) kolaje modelovanej vrcholom, z ktorého hrana vychádza, s cieľom dosiahnuť ďalšiu kolaj. Reverzné hrany vyjadrujú skutočnosť, že vlak môže zmeniť smer pohybu (vykonať reverziu) na kolaji, ktorá je modelovaná počiatočným vrcholom hrany.

Každej hrane grafu  $G_1$  odpovedá dvojica tranzitných hrán v grafe  $G_2$ . Otázkou zostáva, ktoré z vrcholov  $v_i^{(1)}$ ,  $v_i^{(2)}$ ,  $v_j^{(1)}$ ,  $v_j^{(2)}$ , grafu  $G_2$  budú počiatočnými a ktoré koncovými vrcholmi tranzitných hrán, keď v grafe  $G_1$  existuje hrana medzi  $v_i$  a  $v_j$ . Odpoveď nájdeme v tabuľke 2.1. V prvom stĺpci tabuľky je zaznamenané, s ktorými pólmi vrcholov  $v_i$  a  $v_j$  grafu  $G_1$  je hrana incidentná. V druhom stĺpci tabuľky je potom pre každý variant uvedená odpovedajúca dvojica tranzitných hrán v  $G_2$ . Tretí stĺpec tabuľky ukazuje spôsob konštrukcie reverzných hrán. Reverzné hrany môžu vychádzať len z vrcholov, ktoré reprezentujú kolaje (nie výhybkové kolaje). Pritom sa predpokladá, že celý vlak stojí pri reverzii na jednej kolaji. V grafe  $G_2$  nie je modelovaná situácia, že vlak môže pri reverzii obsadiť viacero kolají. Pre každú tranzitnú

Each vertex  $v_i$  of graph  $G_1$  is in graph  $G_2$  represented by a couple of vertices  $v_i^{(1)}$ ,  $v_i^{(2)}$ . It means that a track can be passed in two opposite directions. Edges of  $G_2$  can be divided into *transit* edges (drawn by solid lines) and *reverse* edges (drawn by dotted lines). A transit edge expresses the possibility of a train transit through the track (modelled by the initial vertex of the edge) with the aim of reaching the next track. A reverse edge enables a train to change the direction of its movement (to make a reverse) on the track that is modelled by the initial vertex of the edge.

Each edge of  $G_1$  corresponds to a couple of transit edges of  $G_2$ . However, there is a problem. Let us consider an edge between  $v_i$  and  $v_j$  in  $G_1$ . Which vertex  $v_i^{(1)}$ ,  $v_i^{(2)}$ ,  $v_j^{(1)}$ ,  $v_j^{(2)}$  of graph  $G_2$  will be the initial and terminal vertices of transit edges? The table 2.1 gives the answer. The poles of vertices  $v_i$  and  $v_j$  of graph  $G_1$ , to which the edge is incident, are shown in the first column of the table. For each variant, a corresponding coupling of transit edges of  $G_2$  is written in the second column. The third column of the table shows the reverse edge construction. A reverse edge can just come out from the vertex, which represents the track (not switch track) on which the whole train can reverse. Graph  $G_2$  does not model the situation where a train can occupy more tracks making a reverse. For each transit edge with a terminal vertex  $v_j^{(1)}$



Obr. 2.5. Digraf  $G_2$

Fig. 2.5 Digraph  $G_2$

hranu, ktorej koncovým vrcholom je alebo  $v_j^{(1)}$ ,  $v_j^{(2)}$  vytvoríme reverznú hranu podľa pravidiela v tab. 2.1. Napr. k tranzitnej hrane  $(v_{I'}^{(2)}, v_5^{(2)})$  zostrojíme reverznú hranu  $(v_5^{(2)}, v_{I'}^{(1)})$ .

Transformácia hrán grafu  $G_1$  na hrany grafu  $G_2$

Tab. 2.1

Graf $G_1$ $v_i$ $v_j$	Tranzitné hrany v digrafe $G_2$	Reverzné hrany v digrafe $G_2$
+      -	$v_i^{(1)} \rightarrow v_j^{(1)}$	$v_j^{(1)} \rightarrow v_i^{(2)}$
	$v_j^{(2)} \rightarrow v_i^{(2)}$	
-      +	$v_i^{(2)} \rightarrow v_j^{(2)}$	$v_j^{(2)} \rightarrow v_i^{(1)}$
	$v_j^{(1)} \rightarrow v_i^{(1)}$	
+      +	$v_i^{(1)} \rightarrow v_j^{(2)}$	$v_j^{(2)} \rightarrow v_i^{(2)}$
	$v_j^{(1)} \rightarrow v_i^{(2)}$	
-      -	$v_i^{(2)} \rightarrow v_j^{(1)}$	$v_j^{(1)} \rightarrow v_i^{(1)}$
	$v_j^{(2)} \rightarrow v_i^{(1)}$	

or  $v_j^{(2)}$ , we construct a reverse edge according to the rule given in Tab. 2.1. For example, we construct the reverse edge  $(v_5^{(2)}, v_{I'}^{(1)})$  which is based on the existence of transit edge  $(v_{I'}^{(2)}, v_5^{(2)})$ .

Transformation of edges of  $G_1$  to edges of  $G_2$

Tab. 2.1

Graph $G_1$ $v_i$ $v_j$	Transit edges of digraph $G_2$	Reverse edges of digraph $G_2$
+      -	$v_i^{(1)} \rightarrow v_j^{(1)}$	$v_j^{(1)} \rightarrow v_i^{(2)}$
	$v_j^{(2)} \rightarrow v_i^{(2)}$	
-      +	$v_i^{(2)} \rightarrow v_j^{(2)}$	$v_j^{(2)} \rightarrow v_i^{(1)}$
	$v_j^{(1)} \rightarrow v_i^{(1)}$	
+      +	$v_i^{(1)} \rightarrow v_j^{(2)}$	$v_j^{(2)} \rightarrow v_i^{(2)}$
	$v_j^{(1)} \rightarrow v_i^{(2)}$	
-      -	$v_i^{(2)} \rightarrow v_j^{(1)}$	$v_j^{(1)} \rightarrow v_i^{(1)}$
	$v_j^{(2)} \rightarrow v_i^{(1)}$	

V grafe  $G_2$  sú ohodnotené vrcholy i hrany. Ohodnotenie vrcholu odpovedá dĺžke koľaje, ktorú daný vrchol reprezentuje, pričom platí, že ohodnotenie vrcholov  $v_i^{(1)}$  a  $v_i^{(2)}$  je zhodné. Ohodnotenie hrán je nasledujúce:

- ohodnotenie tranzitnej hrany odpovedá dĺžke koľaje, ktorá je modelovaná počiatočným vrcholom hrany;
- ohodnotenie všetkých reverzných hrán je zhodné a odpovedá dĺžke objektu premeštenia. (Ohodnotenie reverzných hrán teda nie je hranám trvalo priradené v modeli koľajiska, ale určí sa až na začiatku algoritmu pre vyhľadanie najkratšej cesty.)

Najkratšiu trasu pre premeštenie vlaku z počiatočnej na cieľovú koľaj chápeme v grafe ako najkratšiu cestu medzi dvoma vrcholmi. Na jej výpočet môžeme použiť po malých úpravách ľubovoľný algoritmus, ktorý vyhľadá najkratšiu cestu zo počiatočného do koncového vrcholu. Môžeme použiť napr. Dijkstrov algoritmus, ktorý však treba upraviť tak, aby rešpektoval

1. dĺžku vlaku, ktorý chceme premeštniť,
2. aktuálne obsadenie koľajiska.

Obidve požiadavky sú parametrami algoritmu a v priebehu výpočtu sa neaktualizujú. Ako už bolo uvedené vyššie, podľa dĺžky vlaku sa ohodnotia reverzné hrany, takže dĺžka najkratšej cesty bude potom odpovedať skutočnej vzdialnosti, ktorú vlak prejde. Obsadenie koľajiska sa reprezentuje ako vektor reálnych čísel  $o$ . Každý prvok vektora odpovedá jednému vrcholu grafu  $G_2$  a udáva, aká je voľná kapacita koľaje reprezentovanej daným vrcholom (od príslušného konca koľaje). Ak je koľaj celá voľná, sú prvky v ektore  $o_{i1}$  a  $o_{i2}$  odpovedajúce vrcholom  $v_i^{(1)}$  a  $v_i^{(2)}$  zhodné a rovné dĺžke koľaje. Ak je celá koľaj obsadená, sú prvky  $o_{i1}$  a  $o_{i2}$  rovné nule. Ak už na koľaji stojí vlak, ale časť koľaje je voľná, majú prvky vektora  $o_{i1}$  a  $o_{i2}$  hodnotu odpovedajúcu voľnej dĺžke koľaje (pozri obr. 2.6). Dĺžka vlaku a voľná kapacita koľají ovplyvní značkovanie následníkov aktuálneho vrcholu: následník aktuálneho vrcholu sa môže zaradiť do množiny dočasne označených vrcholov len vtedy, ak je jeho voľná kapacita väčšia alebo rovná dĺžke vlaku (vlak sa vojde na nasledujúcu koľaj). Navyše, ak je následník koncovým vrcholom tranzitnej hrany, musí byť voľná kapacita aktuálneho vrcholu rovná ohodnoteniu danej tranzitnej hrany, čo znamená, že vlak môže prejsť cez koľaj, na ktorej sa práve nachádza.

O najkratšej ceste z jedného počiatočného do jedného cieľového vrcholu hovoríme vtedy, ak je presne určený koniec koľaje, ktorým má vlak počiatočnú koľaj opustiť, a tiež je presne určený koniec cieľovej koľaje, cez ktorý má vlak na cieľovú koľaj prísť. Tieto situácie je dôležité rozlišovať najmä v prípadoch, kedy premeštným objektom nie je vlak, ale len lokomotíva. Ked' je na počiatočnej koľaji lokomotíva pripojená k vlaku, nemôže z počiatočnej koľaje odísť cez jej ľubovoľný koniec. Podobne, ak na cieľovej koľaji stojí súprava a chceme k nej prisunúť lokomotív, musíme špecifikovať, ku ktorému koncu

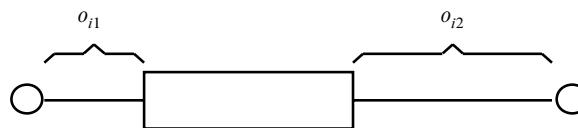
All vertices and edges of  $G_2$  are weighted. The vertex weight corresponds to the length of the track represented by the vertex. The weight of vertex  $v_i^{(1)}$  equals to the weight of vertex  $v_i^{(2)}$ . The edge weights are as follows:

- The transit edge weight corresponds to the length of the track represented by the initial vertex of the edge;
- All reverse edge weights are the same and correspond to the length of the object of transfer. (Therefore, the reverse edge weights are not permanent, but they are determined at the beginning of the procedure for finding the shortest path.)

The shortest route for a train transfer from an initial to a terminal track is understood as the shortest path between a pair of vertices of graph  $G_2$ . An arbitrary algorithm (after a little modification) that finds the shortest path from an initial to a terminal vertex can calculate it. For example, Dijkstra's algorithm can be used supposing it is modified to respect

1. the length of a train to be transferred,
2. the actual trackage occupation.

Both requirements are parameters of the algorithm and they are not actualised during the calculation. As was stated in the preceding section, reverse edges are weighted according to the length of the train. Therefore, the length of the shortest path corresponds to the real distance passed by the train. The trackage occupation is represented by a vector  $o$  of real numbers. Each element of  $o$  corresponds to one vertex of graph  $G_2$  and explains the free capacity of the track represented by this vertex (regarding the appropriate end of the track). If the whole track is free, the vector elements  $o_{i1}$  and  $o_{i2}$  corresponding to  $v_i^{(1)}$  and  $v_i^{(2)}$  are identical and equal to the length of the track. If the whole track is occupied,  $o_{i1}$  and  $o_{i2}$  equal zero. If a train occupies the track, but a part of the track is free, values  $o_{i1}$  and  $o_{i2}$  correspond to the lengths of the free parts of the track (see Fig. 2.6). The length of the train and the free capacities of tracks influence the labelling of successors of the actual vertex. A successor of the actual vertex can be labelled and then added to a search tree if its free capacity is greater or equal to the length of the train (the train fits in at the next track). In addition, if a successor is a terminal vertex of a transit edge, the free capacity of the actual vertex has to equal the weight of that transit edge, which means that a train can pass through the track on which it stands.



Obr. 2.6. Voľná kapacita obsadenej koľaje  $K_i$   
Fig. 2.6 Free capacity of occupied track  $K_i$

The shortest path from one defined initial vertex to one defined terminal vertex is computed, if it is precisely determined the endpoint of the track through which a train shall leave the initial track, as well as it is precisely determined the endpoint of the terminal track, through which a train shall come to the terminal

track. It is important to specify the endpoints of the tracks especially in the case where the object of transfer is not a train, but a locomotive. If a locomotive joins a train on the initial track, it cannot leave the initial track through its arbitrary endpoint. Likewise, if the train cars stand on the terminal track and we want

vlaku ju chceme pripojiť, tzn. z ktorého konca má na cieľovú koľaj príst. Naviac môže mať zmysel aj situácia, kedy má lokomotíva príst na rovnakú koľaj, na ktorej stojí, ale z opačného konca. Ak nepotrebuješ rozlišovať jednotlivé konce počiatocnej koľaje, potom nechladáme cestu z jedného počiatocného vrcholu, ale z množiny počiatocných vrcholov tvorenej vrcholmi  $v_i^{(1)}$  a  $v_i^{(2)}$ . Podobne, ak nešpecifikujeme koniec cieľovej kolaje, uvažujeme dvojprvkovú množinu cieľových vrcholov.

Najkratšiu cestu medzi dvoma vrcholmi grafu  $G_2$  možno ľahko pretransformovať na najkratší sled medzi dvoma hranami grafu  $G_0$ .

### 3. Model koľajiska založený na hranovo ohodnotenom grafe

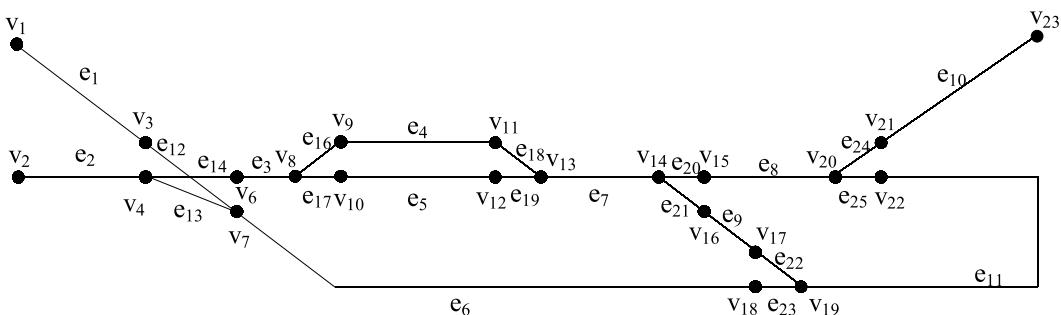
Hranovo ohodnotený graf predstavuje prirodzený model koľajiska, v ktorom hrany reprezentujú kolaje a výhybkové kolaje a vrcholy predstavujú spojenia medzi koľajami. Ohodnenie hrany odpovedá dĺžke koľaje. Na obr. 3.1 je graf  $G$  modelujúci koľajisko z obr. 2.1. Polovičná križovatková (anglická) výhybka sa modeluje bez deliaceho vrcholu v strede výhybky. Hrany, ktoré modelujú výhybku, definujú technicky možné prechody medzi koľajami, ktoré výhybka spája. Polovičná anglická výhybka na obr. 2.1 neumožňuje priamu jazdu z kolaje (hrany)  $e_1$  na kolaj  $e_3$ , preto v grafe  $G$  neexistuje hrana medzi vrcholmi  $v_3$  a  $v_6$ .

to shunt a locomotive to them, we have to determine which endpoint of a train we want it to join (i.e. from which endpoint it shall come to on the terminal track). It might even be the case that a locomotive has to come to the same track on which it stands, but through the opposite endpoint. If the individual endpoints of the initial track need not be distinguished, then we do not search for a path from one initial vertex, but from a set of initial vertices consisting of vertices  $v_i^{(1)}$  and  $v_i^{(2)}$ . Likewise, if the endpoint of the terminal track is not determined, a set of two terminal vertices is taken into account.

The shortest path between a pair of vertices of graph  $G_2$  can be easily transformed to the shortest walk between a pair of edges of graph  $G_0$ .

### 3. A trackage model based upon an edge weighted graph

An edge weighted graph stands for a natural model of a trackage. The edges represent the tracks and the switch tracks. The vertices represent the connections between tracks. The edge weight corresponds to the length of the track. In Fig. 3.1 graph  $G$  models the trackage from Fig. 2.1. A single-slip switch is modelled without a divided vertex in the middle of a switch. The edges modelling a switch define technically allowed transits between tracks connected by a switch. The single-slip switch in Fig. 2.1 does not allow direct transit from track (edge)  $e_1$  to track  $e_3$ , that's why an edge between  $v_3$  and  $v_6$  does not exist in graph  $G$ .



Obr. 3.1 Graf  $G$

Fig. 3.1 Graph  $G$

Uvedený spôsob modelovania výhybek však nedefinuje úplne pravidlá prejazdu cez výhybku. Musíme ešte definovať, že vlak nemôže prejsť priamo z jednej výhybkovej kolaje na druhú, ale že musí urobiť reverziu, t. j. prejsť cez výhybku v jednom smere, zastaviť na nasledujúcej kolaji (alebo viacerých kolajach) a až potom sa na výhybku vrátiť. Napr. pri jazde z kolaje  $e_4$  cez vrchol  $v_{11}$  na kolaj  $e_5$  cez vrchol  $v_{12}$  by vlak musel prechádzať cez kolaje  $e_{18}$ ,  $e_7$  a  $e_{19}$ . Preto definujeme množinu zakázaných odbočení. Jej prvками sú neusporiadané dvojice hrán. Hrany z jednej dvojice modelujú kolaje jednej výhybky. Každá dvojica teda udáva, že nie je možný priamy prechod z jednej výhybkovej kolaje na druhú, ktorá je súčasťou rovnakej výhybky. Pre jednoduchú výhybku vystačíme s jednou neusporiadanou dvojicou, napr.  $[e_{18}, e_{19}]$ . U polovičnej anglickej výhybky musíme definovať dve neusporia-

This way of a switch modelling does not completely define the rules of transit through a switch. We have to define that a train cannot transit directly from one switch track to another, but it has to make a reverse. It means, in fact, that it has to transit through a switch in one direction, to stop on the next track (on more tracks, respectively) and after that return back to a switch. For example, moving from track  $e_4$  through vertex  $v_{11}$  to track  $e_5$  through vertex  $v_{12}$ , a train has to transit through tracks  $e_{18}$ ,  $e_7$  and  $e_{19}$ . That is the reason to define a set of forbidden turns. The elements of the set are represented by unordered pairs of edges. The edges of one pair model tracks of one switch. Therefore, each pair expresses the fact that a direct transit between two tracks which belong to the same switch is not possible. For a simple switch, one pair is sufficient, e.g.  $[e_{18}, e_{19}]$ . For a single-slip

dané dvojice, napr. pre polovičnú križovatkovú výhybku na grafe  $G$  by to boli dvojice  $[e_{12}, e_{13}]$  a  $[e_{13}, e_{14}]$ . Pre celú anglickú výhybku by sme potom definovali štyri neusporiadane dvojice hrán.

Najkratšou cestou pre premiestnenie vlaku z jednej koľaje na druhú je v grafe  $G$  najkratší prípustný sled z počiatočnej na cieľovú hranu. Prípustným sledom rozumieme sled, v ktorom

- žiadne dve po sebe idúce hrany netvoria zakázané odbočenie;
- súčet ohodnotení hrán, ktoré sa v slede opakujú a predstavujú koľaje, na ktorých vlak vykoná reverziu, je väčší alebo rovný dĺžke vlaku.

Najkratší prípustný sled môžeme vyhľadať pomocou Dijkstrovo algoritmu [1, 3], ale musíme ho modifikovať tak, aby rešpektoval zakázané odbočenia na výhybkách, dĺžku premiestňovaného objektu a aktuálne obsadenie koľajiska.

Pretože Dijkstrov algoritmus je vrcholovo orientovaný (hľadá vzdialenosť do vrcholov grafu), musíme definovať, ako sa zakázané odbočenia prejavia vo vrcholovom okolí každého vrcholu. Predovšetkým, vrcholové okolie vrcholu definujeme vzhľadom na každú hranu, s ktorou je vrchol incidentný. Ak teda hrana  $e_l$  má krajné vrcholy  $v_k$  a  $v_z$ , potom vrcholové okolie vrcholu  $v_k$  vzhľadom ku hrane  $e_l$  bude množina obsahujúca všetky jeho susedné vrcholy okrem vrcholu  $v_z$ . Vrcholové okolie vrcholu  $v_k$  vzhľadom na hranu  $e_l$  môžeme ďalej rozdeliť na *tranzitné* a *reverzné* vrcholové okolie. V tranzitnom vrcholovom okolí vrcholu  $v_k$  sa nachádzajú také jeho susedné vrcholy  $v_t$ , že hrany incidentné s týmito vrcholmi a s vrcholom  $v_k$  netvoria s hranou  $e_l$  zakázané odbočenie. Reverzné vrcholové okolie vrcholu  $v_k$  je buď prázdne, alebo je tvorené jedným vrcholom  $v_r$ , ktorý je susedný s vrcholom  $v_k$  a hrana incidentná s vrcholom  $v_r$  a s vrcholom  $v_k$  je v zakázanom odbočení s hranou  $e_l$ . Pre ilustráciu sa vrátme k obr. 3.1 a definujme vrcholové okolie vrcholu  $v_8$  vzhľadom na hranu  $e_{17}$ . Vrcholové okolie vrcholu  $v_8$  vzhľadom na hranu  $e_{17}$  je množina  $\{v_6, v_9\}$ , pričom  $\{v_6\}$  tvorí tranzitné a  $\{v_9\}$  reverzné vrcholové okolie vrcholu  $v_8$  vzhľadom na hranu  $e_{17}$ .

Rozdelenie vrcholového okolia vrcholu  $v_k$  na tranzitné a reverzné má význam pre značkovanie následníkov aktuálneho vrcholu. Ak následník aktuálneho vrcholu  $v_k$  je z reverzného vrcholového okolia, môžeme ho označovať len vtedy, ak za vrcholom  $v_k$  existuje dostatočne dlhá cesta v pôvodnom smere pre zastavenie vlaku. Ak takáto cesta existuje, hovoríme, že vrchol  $v_k$  je dostupný, v opačnom prípade je vrchol  $v_k$  nedostupný a nemôžeme z neho pokračovať po hrane, ktorá je v zakázanom odbočení s hranou, po ktorej sme sa do vrcholu  $v_k$  dostali. V inicializačnej fáze algoritmu sú všetky vrcholy označené ako nedostupné. Po nájdení trasy  $T_u = \{v_p, u_1, u_2, \dots, u_N, u\}$  z počiatočného vrcholu  $v_p$  do vrcholu  $u$  označíme za dostupné všetky vrcholy  $u_i$  na tejto ceste, pre ktoré platí  $d_u - d_{ui} \geq l_O$ , kde  $d_{ui}$  resp. je vzdialosť zo počiatočného vrcholu  $v_p$  do vrcholu  $u$ , resp.  $d_u$  do vrcholu  $u_i$  a  $l_O$  je dĺžka vlaku (objektu premiestnenia).

Následník aktuálneho vrcholu  $v_k$  z jeho reverzného vrcholového okolia nemôžeme teda označovať, ak vrchol  $v_k$  nie je dostupný. Vrchol, ktorý je nedostupný z dôvodu ukončenia alebo obsadenia koľajiska, musíme vrátiť do množiny neoznačených vrcholov, aby ho algoritmus mohol v ďalších krokoch označiť z iného smeru. Situácia je ilustrovaná na obr. 3.2.

switch, two unordered pairs have to be defined, e.g.  $[e_{12}, e_{13}]$  and  $[e_{13}, e_{14}]$ . For a double-s lip switch, four unordered pairs of edges are defined.

The shortest admissible walk from an initial to a terminal edge of graph  $G$  represents the shortest route for train transfer from an initial to a terminal track. An admissible walk is a walk in which

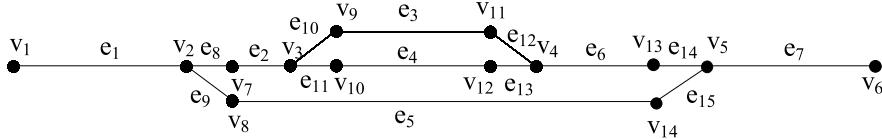
- none of two successive edges belongs to the set of forbidden turns;
- the sum of the weights of edges repeated in a walk and representing tracks, on which a train makes a reverse, is greater or equal to the train length.

The shortest admissible walk can be found by Dijkstra's algorithm [1, 3] which is supposed to be modified so that it could respect forbidden turns on switches as well as the length of an object of transfer and the actual trackage occupation.

The Dijkstra's algorithm uses a vertex-oriented approach (it searches paths to *vertices* of a graph), therefore, we have to define how to express the forbidden turns in a vertex neighbourhood. First, we define a vertex neighbourhood with regard to every edge incident to the vertex. Therefore, if edge  $e_l$  has endpoints  $v_k$  and  $v_z$  then a neighbourhood of  $v_k$  with regard to edge  $e_l$  is a set of its neighbours with the exception of vertex  $v_z$ . A neighbourhood of  $v_k$  with regard to edge  $e_l$  can be divided into a *transit* and a *reverse* neighbourhood. A transit neighbourhood of  $v_k$  consists of those of its neighbours  $v_t$  that the edges with endpoints  $v_t$  and  $v_k$  do not constitute forbidden turns with edge  $e_l$ . A reverse neighbourhood of  $v_k$  is either empty or consists of one vertex  $v_r$  that is a neighbour of  $v_k$  and the edge with endpoints  $v_r$  and  $v_k$  constitutes a forbidden turn with edge  $e_l$ . For an illustration, let us go back to Fig. 3.1 and define the neighbourhood of vertex  $v_8$  with regard to edge  $e_{17}$ . The neighbourhood of vertex  $v_8$  with regard to edge  $e_{17}$  is set  $\{v_6, v_9\}$ , whereby  $\{v_6\}$  stands for the transit and  $\{v_9\}$  stands for the reverse neighbourhood of vertex  $v_8$  with regard to edge  $e_{17}$ .

The division of a neighbourhood of vertex  $v_k$  to a transit and a reverse one is important for the labelling of successors of the actual vertex. If a successor of the actual vertex  $v_k$  belongs to its reverse neighbourhood, it can be labelled only if a route (behind the vertex  $v_k$  in the original direction) is long enough for a train stop. If the required route exists, the vertex  $v_k$  is said to be accessible. Otherwise, the vertex  $v_k$  is inaccessible and the route cannot go on using an edge, which constitutes a forbidden turn with the edge coming into vertex  $v_k$ . During the initialisation phase of the algorithm, all vertices are indicated as inaccessible. When the route  $T_u = \{v_p, u_1, u_2, \dots, u_N, u\}$  from initial vertex  $v_p$  to vertex  $u$  was found, all vertices  $u_i$  on this path which holds  $d_u - d_{ui} \geq l_O$ , are indicated as accessible. We denoted the distance from the initial vertex  $v_p$  to the vertices  $u$  and  $u_i$  by  $d_u$  and  $d_{ui}$  respectively and the length of an object of transfer by  $l_O$ .

Therefore, a successor of the actual vertex  $v_k$  from its reverse neighbourhood cannot be labelled until  $v_k$  is accessible. A vertex that is inaccessible due to the end of trackage or the track occupation has to be deleted from a search tree so that the algorithm could label it in the next search process through another predecessor. The situation is illustrated in Fig. 3.2.



Obr. 3.2. Hľadanie alternatívnej cesty pri obmedzenej dĺžke koľajiska  
Fig. 3.2 Searching of an alternative path in a restricted trackage

Vlak, ktorý stojí na koľaji  $e_5$ , chceme premiestniť na koľaj  $e_4$ , pričom požadujeme, aby vlak prišiel na koľaj  $e_4$  zľava. Dĺžka koľaje  $e_1$  je menšia než dĺžka vlaku a neumožňuje reverziu pohybu, preto vlak nemôže pribúť na koľaj  $e_4$  najkratšou cestou cez koľaje  $e_9$ ,  $e_1$ ,  $e_8$ ,  $e_2$  a  $e_{11}$ . Súčet dĺžok koľají  $e_1$ ,  $e_8$  a  $e_2$  je naopak dostačočný a cesta cez koľaj  $e_3$  s reverziou na koľajach  $e_2$ ,  $e_8$  a  $e_1$  je prípustná. Aby algoritmus toto prípustné riešenie našiel, musí po neúspešnom pokuse o reverziu vo vrchole  $v_2$  (pri ceste zo koľaje  $e_5$ ) nedostupné vrcholy  $v_1$  a  $v_2$  uvoľniť pre hľadanie cesty z iného smeru. To znamená, že označenie vrcholov sa nastaví na počiatkovú hodnotu ( $d_1 = \infty$  a  $d_2 = \infty$ ). Okrem toho, po preznačení nedostupného vrcholu sa do množiny dočasne označených vrcholov musí zaradiť taký jeho susedný vrchol, ktorý nie je jeho predchodom, ale má značku  $< \infty$ , pretože tento vrchol sa môže stať predchodom nedostupného vrcholu na novej ceste.

Algoritmus musí ďalej rešpektovať aktuálne obsadenie koľiska. Obsadenie koľiska je vyjadrené ako voľná kapacita koľaja vztiahnutá ku každému koncu koľaje (obr. 2.6). Podobne ako v predchádzajúcim modeli, dĺžka vlaku a obsadenie koľiska sú vstupnými parametrami algoritmu.

#### 4. Výpočtové experimenty

Obidva spôsoby modelovania koľiska a na nich založené algoritmy pre výpočet najkratšej jazdnej cesty sme testovali na ilustračnom koľisku z obr. 2.1 a na reálnom koľisku zriaďovacej stanice Žilina - Teplička nad Váhom (obr. 4.1). Cieľom výpočtových experimentov bolo porovaňa obidva spôsoby modelovania koľiska z hľadiska rýchlosťi výpočtu najkratšej jazdnej cesty.

Na digrafe z obr. 2.5 sme nechali algoritmus vypočítať najkratšie cesty medzi všetkými dvojicami vrcholov, a to pre rôzne dĺžky vlaku (v rozsahu od 0 do 500 m). Pre každú dĺžku vlaku sme potom vypočítali priemerný čas pre vyhľadanie jednej cesty.

Digraf modelujúci koľisko zriaďovacej stanice Žilina - Teplička nad Váhom má 1178 vrcholov a 1095 hrán. Na takom rozsiahлом modeli by bolo časovo náročné (a asi aj zbytočné) počítať všetky najkratšie cesty, preto algoritmus počíta najkratšie cesty len medzi prvými 200 vrcholmi.

Všetky výpočty prebiehali za predpokladu, že koľisko je prázdne (všetky koľaje sú voľné).

Výsledky experimentov na oboch koľiskách sú zhrnuté v tab. 4.1. V prvých dvoch stĺpcoch tabuľky je počet vrcholov, resp. počet hrán digrafa modelujúceho dané koľisko. Tretí stĺpec

A train standing on track  $e_5$  shall be transferred to track  $e_4$ . We want it to come to the terminal track through its left endpoint. The length of track  $e_1$  is less than the length of the train and it does not allow for making a reverse. That's why the train cannot come to  $e_4$  using the shortest route through tracks  $e_9$ ,  $e_1$ ,  $e_8$ ,  $e_2$  and  $e_{11}$ . On the other hand, the sum of the lengths of tracks  $e_1$ ,  $e_8$  and  $e_2$  is sufficient and the route through track  $e_3$  (with reversion on tracks  $e_2$ ,  $e_8$  and  $e_1$ ) is admissible. So that the algorithm could find this admissible solution, it has to remove inaccessible vertices  $v_1$  and  $v_2$  from a search tree after an unsuccessful reversion behind vertex  $v_2$  (on a path from  $e_5$ ). It means that the labels of  $v_1$  and  $v_2$  are set again to their initial values ( $d_1 = \infty$  a  $d_2 = \infty$ ). In addition, when an inaccessible vertex  $v_k$  was re-labelled, its neighbour having a label and not being a predecessor of  $v_k$  has to be indicated as the temporary labelled vertex in order so that it could become a predecessor of  $v_k$  on a new path.

The algorithm has to respect the actual trackage occupation. The trackage occupation is expressed as the free capacity of each track related to the endpoint of the track (Fig. 2.6). As in the preceding model, the length of the train and the trackage occupation represent input parameters of the algorithm.

#### 4. Computational experiments

Both methods of trackage modelling and algorithms for finding the shortest train route based upon them were tested on the illustrative trackage in Fig. 2.1, as well as on the real trackage of marshalling yard Žilina - Teplička nad Váhom (Fig. 4.1). The goal of computational experiments was to compare both methods of trackage modelling in terms of the time performance of the algorithms.

On the digraph in Fig. 2.5 the algorithm was run for all pairs of vertices and for various train lengths (ranging from 0 to 500 m). Then for each train length an average algorithm calculation time was measured.

The digraph, modelling the trackage of marshalling yard Žilina - Teplička nad Váhom, is composed of 1178 vertices and 1095 edges. It would be time-consuming (and probably useless) to find all the shortest paths on such a large model. That's why the algorithm was run just for pairs of the first 200 vertices.

All computations were done supposing the trackage is empty (all tracks are free).

The computational results for both track infrastructures are summarised in Table 4.1. The number of vertices and the number of edges of the digraph (modelling the given trackage) are in the first two columns of the table. The third column labelled  $N_c$

tabuľky označený symbolom  $N_c$  udáva počet vrcholov, medzi ktorými by sme chceli vypočítať najkratšie cesty. Pred spustením výpočtu sa najprv skontroluje, či výpočet má zmysel, t. j. či dĺžka kolají modelovaných počiatocným a cieľovým vrcholom je väčšia alebo rovná dĺžke vlaku. Počet cest, ktoré algoritmus počíta, teda závisí od dĺžky vlaku a platí preň vzťah počet cest  $\leq N_c^2 - N_c$ . Do tohto počtu sú zahrnuté aj také cesty, ktoré algoritmus začal počítať, ale nedopocítal, pretože zistil, že požadovaná cesta neexistuje.

Ďalší stĺpec tabuľky obsahuje celkový čas výpočtu všetkých cest. V poslednom stĺpci je priemerný čas výpočtu jednej cesty.

Výsledky výpočtových experimentov pre ohodnotený digraf Tab. 4.1

N	M	$N_c$	Dĺžka vlaku [m]	Počet cest	Celkový čas [s]	Priemerný čas [ms]
50	78	50	0	2450	1.10	0.45
50	78	50	50	380	0.22	0.58
50	78	50	100	306	0.17	0.56
50	78	50	200	132	0.11	0.83
50	78	50	300	30	0.06	2.00
50	78	50	400	12	0.05	4.17
50	78	50	500	12	0.06	5.00
1178	1905	200	0	39800	578.64	14.54
1178	1905	200	50	5402	90.24	16.70
1178	1905	200	100	2450	38.56	15.74
1178	1905	200	200	1560	24.66	15.81
1178	1905	200	300	1122	18.90	16.84
1178	1905	200	400	756	13.02	17.22
1178	1905	200	500	756	12.31	16.28

Výsledky výpočtových experimentov pre neorientovaný hranovo ohodnotený graf Tab. 4.2

N	M	$M_c$	Dĺžka vlaku [m]	Počet sledov	Celkový čas [s]	Priemerný čas [ms]
22	24	24	0	2256	1.65	0.73
22	24	24	50	650	0.49	0.75
22	24	24	100	306	0.28	0.92
22	24	24	200	132	0.11	0.83
22	24	24	300	30	0.06	2.00
22	24	24	400	12	0.05	4.17
22	24	24	500	12	0.05	4.17
690	577	100	0	39800	452.64	11.37
690	577	100	50	5402	72.56	13.43
690	577	100	100	2450	41.31	16.86
690	577	100	200	1560	31.80	20.38
690	577	100	300	1122	27.85	24.82
690	577	100	400	756	24.01	31.76
690	577	100	500	756	24.44	32.33

involves the number of vertices, between which we would like to find shortest paths. Before the algorithm starts running, it is checked to see if the lengths of tracks modelled by the initial and the terminal vertices are greater or equal to the length of the train. Therefore, the number of paths the algorithm searches depends on the length of the train and the following relation holds: *number of paths*  $\leq N_c^2 - N_c$ . The number of paths includes also those paths, which the algorithm started to search, but it fails due to the fact that the required path did not exist.

The next column of the table involves the total computation time of all the paths. There is an average time of one algorithm run in the last column.

Computational results for a weighted digraph

Tab. 4.1

N	M	$N_c$	Train lenght [m]	Number of walks	Total time [s]	Average time [ms]
50	78	50	0	2450	1.10	0.45
50	78	50	50	380	0.22	0.58
50	78	50	100	306	0.17	0.56
50	78	50	200	132	0.11	0.83
50	78	50	300	30	0.06	2.00
50	78	50	400	12	0.05	4.17
50	78	50	500	12	0.06	5.00
1178	1905	200	0	39800	578.64	14.54
1178	1905	200	50	5402	90.24	16.70
1178	1905	200	100	2450	38.56	15.74
1178	1905	200	200	1560	24.66	15.81
1178	1905	200	300	1122	18.90	16.84
1178	1905	200	400	756	13.02	17.22
1178	1905	200	500	756	12.31	16.28

Computational results for a non-oriented edge weighted graph

Tab. 4.2

N	M	$M_c$	Train lenght [m]	Number of walks	Total time [s]	Average time [ms]
22	24	24	0	2256	1.65	0.73
22	24	24	50	650	0.49	0.75
22	24	24	100	306	0.28	0.92
22	24	24	200	132	0.11	0.83
22	24	24	300	30	0.06	2.00
22	24	24	400	12	0.05	4.17
22	24	24	500	12	0.05	4.17
690	577	100	0	39800	452.64	11.37
690	577	100	50	5402	72.56	13.43
690	577	100	100	2450	41.31	16.86
690	577	100	200	1560	31.80	20.38
690	577	100	300	1122	27.85	24.82
690	577	100	400	756	24.01	31.76
690	577	100	500	756	24.44	32.33

Rovnakým spôsobom bol testovaný aj druhý model - neorientovaný hranovo ohodnotený graf. Na modeli ilustračného koľajiska (graf na obr. 3.1) sme pre rôzne dĺžky vlaku vypočítali všetky prípustné sedy. Na modeli koľajiska zriaďovacej stanice Žilina - Teplička nad Váhom sme vypočítali všetky prípustné sedy medzi prvými 100 hranami.

Výsledky testov sú zhrnuté v tab. 4.2. Horná polovica tabuľky plati pre ilustračné koľajisko, dolná polovica pre koľajisko zriaďovacej stanice Žilina - Teplička nad Váhom. V prvých dvoch stĺpcoch tabuľky je počet vrcholov, resp. počet hrán grafu modelujúceho dané koľajisko. Tretí stĺpec tabuľky označený symbolom  $M_c$  udáva počet hrán, medzi ktorými by sme chceli vypočítať najkratšie sedy. Pre počet sledov, ktoré algoritmus počíta (piaty stĺpec), platí vzťah  $\text{počet sledov} \leq 4M_c^2 - 2M_c$ . Pre vlak dĺžky 0 m plati v tomto vzťahu rovnosť a algoritmus vyhľadá vlastne najkratšie cesty. V posledných dvoch stĺpcach tabuľky je celkový čas výpočtu všetkých sledov a priemerný čas výpočtu jedného sledu.

Výpočtové experimenty boli vykonané na osobnom počítači s procesorom I80486DX/100MHz.

Porovnaním výsledkov v tabuľkách 4.1 a 4.2 zistíme, že priemerná doba výpočtu najkratšej jazdnej cesty je pri oboch spôsoboch modelovania koľajiska rádovo rovnaká. Na reálnom koľajisku pre vlak dlhší ako 100 m je výpočet na digrafe o niečo rýchlejší než výpočet na neorientovanom grafe, pričom rozdiel v rýchlosti stúpa so zväčšujúcou sa dĺžkou vlaku.

## 5. Záver

Navrhnuté modely koľajiska a algoritmy výpočtu najkratšej jazdnej cesty rešpektujú špecifický charakter koľajovej siete v železničnej stanici a pravidlá, ktorími sa riadi pohyb koľajových vozidiel. Algoritmy dokážu vypočítať optimálnu trasu pre pre-miestnenie vlaku (o zadanej dĺžke) za plnej prevádzky na stanici, kedy sa v koľajisku nachádzajú súčasne iné vlaky, ktoré obsadzujú alebo blokujú koľaje. Na druhej strane sa algoritmy dajú využiť na vytvorenie databázy cest, ktoré boli vypočítané pre nulovú dĺžku vlaku a na prázdnom koľajisku. V takto vytvorenej databáze vyhľa-dáme cestu v prípadoch, kedy často používame rovnakú cestu, nezáleží na dĺžke vlaku a je zaručené, že všetky koľaje sú voľné. Typickým príkladom takej situácie je triedenie vlakov na zvážnom pahorku. Všetky vozne, ktoré patria do jednej relácie, smerujú zo zvážného pahorku na rovnakú smerovú koľaj. Prechádzajú teda cez rovnaké koľaje, ktoré musia byť voľné. Vyhľadaním cest v databáze sa vyhneme opakovanému výpočtu rovnakej cesty.

Prvý model - ohodnotený digraf - je v porovnaní s druhým modelom - hranovo ohodnoteným neorientovaným grafovom - menej presný. Je preto použiteľný v takých úlohách, kde vystačíme so zjednodušeným modelovaním pohybu koľajových vozidiel (nemodelujeme úplne presne obsadenie koľají pri reverzii). Tento prístup môžeme uplatniť napríklad v simulačnom modeli pre-vádzky železničnej zriaďovacej stanice, ktorý skúma len kapacitné možnosti stanice a nezaujíma sa o presné obsadenie všetkých koľají [7]. Druhý model je veľmi podrobnej. Umožňuje detailné

The second model - a non-oriented edge weighted graph - was tested in the same way. All admissible walks for various train lengths were searched on the model of the illustrative trackage (the graph in Fig. 3.1). On the trackage model of marshalling yard Žilina - Teplička nad Váhom, the admissible walks between the first 100 edges were searched.

The computational results are summarised in Table 4.2. The upper half of the table demonstrates the illustrative trackage, the bottom half the trackage of marshalling yard Žilina - Teplička nad Váhom. The number of vertices and the number of edges of the graph modelling the given trackage are in the first two columns of the table. The third column labelled  $M_c$  involves the number of edges, between which we would like to find shortest walks. For the number of walks the algorithm searches the following relation holds: *number of walks*  $\leq 4M_c^2 - 2M_c$ . For a train of zero length the equality holds in the mentioned relationship and the algorithm finds in fact the shortest paths. The total computation time of all the walks and an average time of one walk are in last two columns of the table.

Computational experiments were done using a personal computer with processor I80486DX/100MHz.

Comparing results in tables 4.1 and 4.2, we find out that an average time for finding the shortest train route is of the same order for both methods of trackage modelling. The algorithm on a digraph outperforms slightly the algorithm on a graph for train length greater than 100 m. The distinction in performance increases with an increase in train length.

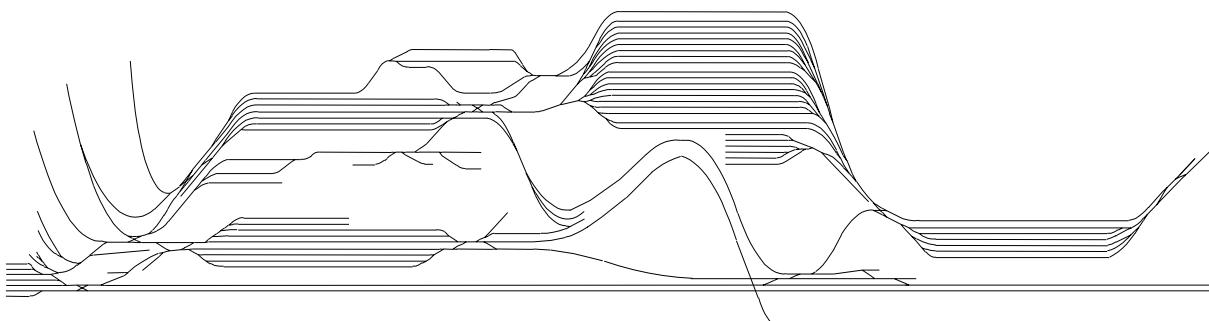
## 5. Conclusions

The designed trackage models and the algorithms for finding the shortest train route respect a specific character of the track network in a railway node, as well as the rules for the movement of rail vehicles. The algorithms are able to find an optimal route for a transfer of a train (not neglecting its length) under the complex marshalling yard operation when a lot of other railway vehicles occupy or block the trackage. On the other hand, the algorithms can be used to create a database of routes being calculated for a train of zero length and for an empty trackage. Then we can query this database in case the same route is often needed when it does not depend upon the length of the train and all tracks are free. A typical example of this situation is a train sorting on a hump. All train cars of one direction move from a hump to the same sorting track. They pass the same tracks and switches which must be free. Finding such routes in the database, we do not need to repeat the calculation of the same route.

The first model - a weighted digraph - is less precise compared to the second model - a non-oriented edge weighted graph. It is useful in those problems where simplified model of the movement of rail vehicles is sufficient (we do not need to model absolutely precisely the occupation of tracks during the reverse). This approach can be applied, for example, in a simulation model of marshalling yard operation which investigates capacity possibilities of a marshalling yard and is not interested in the precise occupation of all tracks [7]. The second model is very detailed. It

sledovanie pohybu vlaku a verne odráža situáciu v reálnom koľajisku. Na druhej strane, výsledky výpočtových experimentov na reálnom koľajisku ukázali, že algoritmus na digrafe vyhľadá najkratšiu trasu pre premiestnenie vlaku dlhšieho ako 100 m rýchlejšie než upravený Dijkstrova algoritmus na neorientovanom grafe, a to aj napriek tomu, že digraf má oproti neorientovanému grafu zhruba dvojnásobný počet vrcholov a trojnásobný počet hrán.

allows for investigating the movement of a train in detail and faithfully reflects the situation in a real trackage. On the other hand, the computational results on the real trackage showed that the algorithm based upon a digraph outperforms modified Dijkstra's algorithm on a non-oriented graph for train length greater than 100 m, although a digraph has double the number of vertices and triple the number of edges than a non-oriented graph.



Obr. 4.1. Plán železničnej zriaďovacej stanice Žilina - Teplička nad Váhom  
Fig. 4.1 Plan of marshalling yard Žilina - Teplička nad Váhom

## Literatúra

- [1] CENEK, P.: Modelování a optimalizace procesů na dopravních sítích, habilitační práce, Žilinská univerzita, Žilina, 1996
- [2] CENEK, P.: Simulation of processes in a marshalling yard, In: Proceedings of COMPRAIL '96 conference in Berlin - Germany, Wessex Institute of Technology-Computational Mechanics Publications, Southampton - UK, 1996, pp. 501-510
- [3] CENEK, P., KLIMA, V., JANÁČEK, J.: Optimalizace dopravních a spojových procesů, VŠDS, Žilina, 1994
- [4] KAVIČKA, A.: Aplikace polárních grafov v dopravě, Sborník přednášek z 9. Mezinárodní vědecké konference, VŠDS, Žilina, 1993, str. 191-197
- [5] KAVIČKA, A.: Modelování kolejisti a algoritmy výpočtu nejkratších jízdních cest. Dizertačná práca, Žilinská univerzita, Žilina, 1997
- [6] KLIMA, V., KAVIČKA, A.: Virtual railway marshalling yard, In: Preprints of the „IFAC/IFIP/IFORS Symposium - Transportation systems“, Technical University of Crete, Chania, Greece, 1997, pp. 880-883, vol. 2
- [7] SADLOŇ, L.: Simulačný model železničnej zriaďovacej stanice, kandidátska dizertačná práca, VŠDS, Žilina, 1994
- [8] ZELINKA, B.: Polar graphs and railway traffic, Aplikace matematiky 3/1974, pp. 169-176

Recenzenti: K. Šotek, S. Palíček

## References

- [1] CENEK, P.: Modelování a optimalizace procesů na dopravních sítích, habilitační práce, Žilinská univerzita, Žilina, 1996
- [2] CENEK, P.: Simulation of processes in a marshalling yard, In: Proceedings of COMPRAIL '96 conference in Berlin - Germany, Wessex Institute of Technology-Computational Mechanics Publications, Southampton - UK, 1996, pp. 501-510
- [3] CENEK, P., KLIMA, V., JANÁČEK, J.: Optimalizace dopravních a spojových procesů, VŠDS, Žilina, 1994
- [4] KAVIČKA, A.: Aplikace polárních grafov v dopravě, Sborník přednášek z 9. Mezinárodní vědecké konference, VŠDS, Žilina, 1993, str. 191-197
- [5] KAVIČKA, A.: Modelování kolejisti a algoritmy výpočtu nejkratších jízdních cest. Dizertačná práca, Žilinská univerzita, Žilina, 1997
- [6] KLIMA, V., KAVIČKA, A.: Virtual railway marshalling yard, In: Preprints of the „IFAC/IFIP/IFORS Symposium - Transportation systems“, Technical University of Crete, Chania, Greece, 1997, pp. 880-883, vol. 2
- [7] SADLOŇ, L.: Simulačný model železničnej zriaďovacej stanice, kandidátska dizertačná práca, VŠDS, Žilina, 1994
- [8] ZELINKA, B.: Polar graphs and railway traffic, Aplikace matematiky 3/1974, pp. 169-176

Reviewed by: K. Šotek, S. Palíček