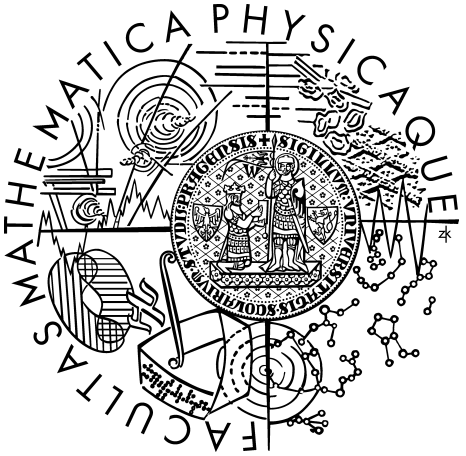
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE



Jakub Uvíra

Užití dynamického programování pro návrh pohybu letadel po ploše letiště.

Katedra aplikované matematiky

Vedoucí diplomové práce: Prof. RNDr. Karel Zimmermann, DrSc.

Studijní program: Informatika, Diskrétní matematika
a optimalizace

2008

Tato práce by nevznikla bez pomoci pracovníků Řízení letového provozu České republiky, kterým tímto děkuji. Jmenovitě Tomáši Singerovi za mnoho informací a předpisů ohledně pohybu letadel nejenom po letištní ploše ale i ve vzduchu. Dále děkuji Raineru Kiehne za návrh tématu této práce, protože bez jeho návrhu by práce nikdy nevznikla. Michalu Markovi za pomoc při vytváření situačních obrázků a mapy letiště. Svým rodičům, příbuzným a přátelům, kteří mně při vytváření této práce podporovali.
Zvláště bych chtěl poděkovat vedoucímu této diplomové práce, kterým byl Prof. RNDr. Karel Zimmermann, DrSc., za mnoho užitečných rad při psaní této práce a několikanásobné pečlivé přečtení celého textu.

Prohlašuji, že jsem svou diplomovou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a jejím zveřejňováním.

V Praze dne 16. dubna 2008 Jakub Uvíra

Obsah

Název práce: Užití dynamického programování pro návrh pohybu letadel po ploše letiště.
Autor: Jakub Uvíra
Katedra (ústav): Katedra aplikované matematiky
Vedoucí diplomové práce: Prof. RNDr. Karel Zimmermann, DrSc.
e-mail vedoucího: Karel.Zimmermann@mff.cuni.cz

Abstrakt: V této práci se zabýváme návrhem algoritmu pro optimalizaci pořadí letadel při startu z letiště. Využívají se metody dynamického a lineárního programování. Algoritmy jsou navrhovány tak aby byla možná jejich implementace, která by byla rychlá, a využívala co nejvíce informací které již byly někdy vytvořeny. V první půlce práce se zabýváme úlohou dynamického programování. Druhá půlka je o lineárním programování které řeší případy, do kterých již nemůže úloha dynamického programování zasáhnout. Jsou zde popsány algoritmy na vytvoření těchto úloh, které by následně byly řešeny za pomocí optimalizačního software.

Klíčová slova: Dynamické programování, Letiště, Letadla, Lineární programování, Optimalizace

Title: Usage of dynamic programming for proposing aircraft movements on an airport.
Author: Jakub Uvíra
Department: Department of Applied Mathematics
Supervisor: Prof. RNDr. Karel Zimmermann, DrSc.
Supervisor's e-mail address: Karel.Zimmermann@mff.cuni.cz

Abstract:In this work we deal with the suggestion of the algorithm for the optimization of the sequences of the aeroplanes during the start from the airport. The dynamic programming and the linear programming been applying. Algorithms are suggested in such a way that its implementation would be possible for the quick use and that would be using the most information which have been already created. In first half of the work we cover with the task of the dynamic programming. The second half is about the linear programming which solves the conditions in which the task of the dynamic programming cannot be involved. The algorithms for creating this works are described here to be consequently solved by the help of the optimizing software.

Keywords: Dynamic programming, Airport, Aeroplanes, Linear programming, Optimization

Kapitola 1

Úvod

Tato diplomová práce se věnuje jednomu praktickému problému, který vede na matematickou optimalizaci. Jde o minimalizaci času, který letadla potřebují pro start a přistání na určitém letišti. V German Aerospace Center se zabývají mnoha dalšími problémy, které v leteckém provozu vznikají, ale problém řazení a pojíždění letadel na start zatím nebyl zkoumán. Praktické zkušenosti jsem čerpal od pracovníků Řízení letového provozu České republiky, případně z pozorování přímo na letišti. Předpokládá se, že kdyby se výsledky této práce podařilo aplikovat do praxe, tak by informace získané z vytvořeného programu mohly pomoci pracovníkovi, který má na starost pozici Executive Controller GROUND[1] na TWR[2]. Pro sestavování optimalizační úlohy není důležitá jen část práce pro navigování letadel prováděná z TWR, ale je také důležité brát v úvahu činnost dalších složek Řízení letového provozu. Jmenovitě APP[3] a ACC[4]. Důvody budou vysvětleny v části *2.9 Přechod mezi APP a ACC*. Výsledkem této práce by mělo být určení optimálního pořadí, v jakém mají letadla na daném letišti startovat. Kromě informace o pořadí bude řešení optimalizační úlohy obsahovat informace o času startu jednotlivých letadel. Z tohoto důvodu se práce snaží minimalizovat čas, po který letadlo stojí s nastartovanými motory, protože každá minuta takového stání nejen stojí peníze, ale také znečišťuje ovzduší. Letadla jsou sice z velké části řízena počítačem, ale intervence lidského faktoru stále ovlivňuje rozhodovací proces. Proto se práce sice zabývá přesnými algoritmy, ale všechny měřitelné proměnné budou v sobě zahrnovat i časovou rezervu beroucí v úvahu tento lidský faktor.

Zároveň je v této práci důležité sestavit takovou úlohu dynamického programování, kterou by byl počítač schopen zpracovávat v reálném čase (aby byly výsledky aplikovatelné). Z tohoto důvodu zde budou zohledňována taková řešení daného problému, která lze efektivně naprogramovat.

[1] osoba zodpovědná mimo jiné za pojezd letadel po ploše,
[2] řídící věž letiště, případně oblast ve vzduchu, kde jsou letadla řízena z věže
[3] název oblasti pro řízení přiblížení a odletu z letiště
[4] Většina vzdušného prostoru, kde se řídí přelet letadla

Kapitola 2

Popis provozu na letišti

Většina lidí by si představuje, že letecká doprava je pravidelná a proto stačí, když je daný problém vyřešen jednou pro vždy. Bohužel tomu tak není. Nejenže každý den může pravidelné letadlo odletět v jiný čas, ale zároveň se na letišti objevují i letadla, která odlétají nepravidelně. Jako příklad by se dal uvést odlet nějakého politika. Také u pravidelného letadla nemůže být občas přesně dodržen přidělený čas odletu, zvaný odletový slot[1], protože na cílovém letišti daného letadla může být v plánovanou dobu příletu velký provoz, a kdyby se letadlo zpozdilo nebo přiletělo brzo, tak bude muset čekat dlouho ve vzduchu. Proto musí být, nejen z výše uvedených důvodů, navrhovaný algoritmus používán nepřetržitě. Je nezbytně nutné brát velký zřetel na jeho maximální efektivitu, aby algoritmus rychle nalezl řešení momentální situace.

2.1 Teoretická velikost úlohy

Při tvorbě algoritmu je třeba brát na vědomí velikost úlohy, protože např. na pražském letišti v průběhu letních měsíců odlétá denně více jak 200 letadel. Na největším letišti světa v Atlantě je tento počet dokonce větší než 1300, což znamená jedno letadlo každých 66 sekund po celý den.

Pokud by byla vytvářena úloha obsahující všechna letadla, které se v daný den objevují na daném letišti, tak by úloha nabyla velikosti, kterou by nebylo možné v reálném čase zpracovat. Z tohoto důvodu není vhodné vytvářet úlohu dynamického programování kdy zpracováváme všechny starty jako jeden celek. Je mnohem vhodnější vytvářet malé dynamické úlohy, které budou řešit aktuální problémy. Z ohledu efektivity řešení je vhodné tyto úlohy navrhovat tak, aby pro vyřešení aktuální úlohy dynamického programování bylo využito co nejvíce informací z úlohy předešlé. Z tohoto ohledu se bude úloha vytvářet způsobem, aby tento postup byl možný. Vytváření úloh se bude podobat vytváření úlohy dynamického programování, která by běžela pro celý den. Jen se vždy z daných podmínek bude počítat pouze aktuální úloha v potřebném časovém intervalu.

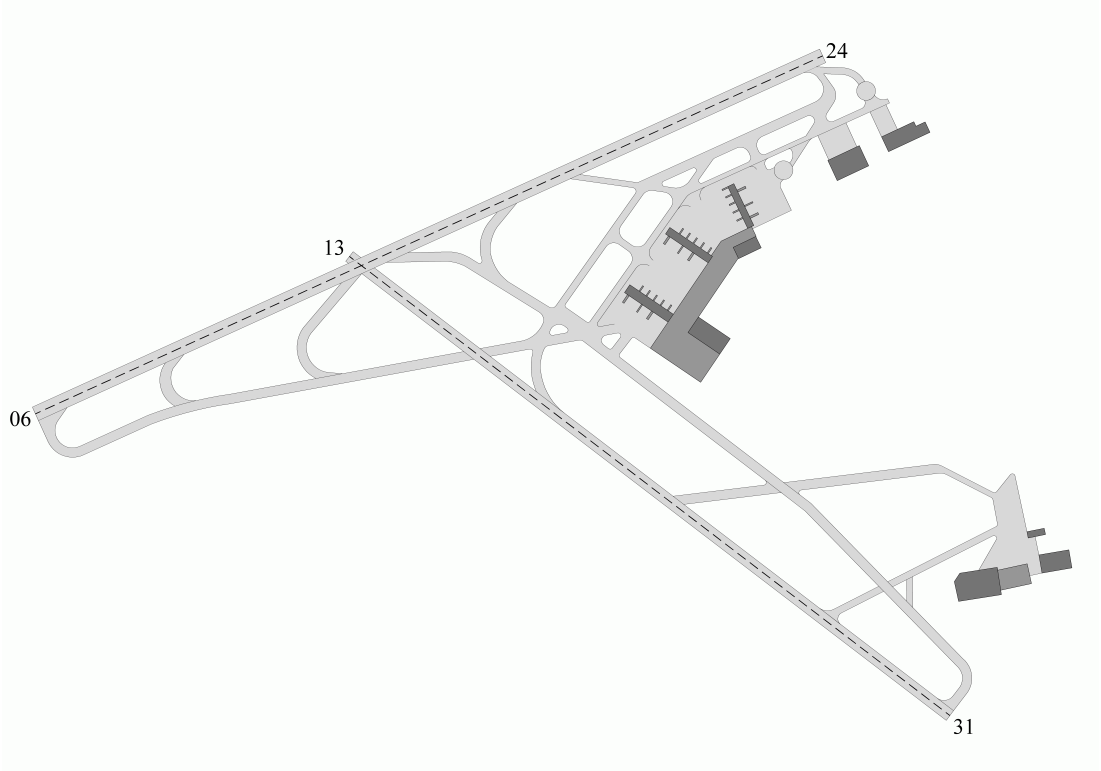
[1] Odletový slot je časový interval, ve kterém letadlo musí odletět. Pokud neodletí,tak se musí zažádat o nový slot

2.2 Základní časová jednotka

Pro veškeré zpracovávání daného problému je vhodné rozdělit si čas, ve kterém problém zkoumáme, diskrétně. Výhodou je, že se následně nemusí úlohy počítat se spojitým časem. V této práci budeme interpretovat tyto diskrétní stavy jako počátky a konce časových intervalů, přičemž pro dva časové intervaly, které následují po sobě platí, že konec prvního se rovná začátku druhého. V této práci budeme tento časový interval označovat jako **základní časovou jednotku**.

2.3 Letiště

Předložená práce bude popisovat algoritmy, které budeme aplikovat na letišti znázorněném na obrázku 2.1. Ve většině případů bude použit jen nejnutnější výřez z tohoto obrázku. V tomto obrázku je u každé ranveje zapsáno její číslo[2], v ostaních obrázcích nebudou čísla ranvejí uváděna, aby čtenáře zbytečně nemátla. V práci budeme vždy jako používanou dráhu brát dráhu číslo 24.



Obrázek 2.1: Plán letiště

[2]ve většině textu je využito číslo 24. Číslo ranveje je úhel ve stupních, který svírá směr ranveje se směrem poledníků, vydělený desíti a zaokrouhlený na celé číslo. Přesnou definici lze najít v[6]

2.4 Definice popisu letadel

V každé úloze, kterou bude tento algoritmus generovat, budou běžná letadla označena jako $L_1, \ldots, L_h$, kde $h \in N, h$ značí počet letadel v dané úloze. Není nutno označovat letadlo písmenem L, protože letadla jsou odlišována za čísla v dolním indexu, pokud bychom ale používali jen číslo, tak by se úloha mohla stát méně přehlednou. V práci se budou také vyskytovat letadla značená jako L_0, L_{h+1}. Tyto letadla jsou oproti ostatním letadlům něčím zvláštní. Proto jsou označována samostatně.

Úlohou L^i budeme značit takovou úlohu, ve které se vyskytují výše popsaná letadla $L_0, \ldots, L_{h+1}$. Úloha nemusí obsahovat všechna výše uvedená letadla, ale u každé definice úlohy bude uvedeno, která letadla úloha obsahuje. Úlohu můžeme též označovat jen písmenem L. Horní index bude použit jen v případech, kdy se v příkladě bude paralelně vyskytovat více úloh. Jako ukázku takového příkladu lze například zmínit úlohu, která bude využívat data vypočtená jinou úlohou.

Pro každé letadlo jsou definovány následující proměnné:

Doba pojíždění letadla

Každému letadlu trvá cesta od stojánky k ranveji určitou dobu. Tato doba musí být ve výpočtech zohledněna, protože pojíždění letadel ovlivňuje úlohu zásadním způsobem.
Označme počet základních časových jednotek, kolik minimálně trvá letadlu L_i cesta ze stojánky na start jako $(tc)_i$. Přičemž $(tc)_i$ je vždy celé číslo, $i \in 0, \ldots h+1$.

Směr odletu

Každé letadlo má určený směr odletu z letiště. Tato informace neříká kam následně letadlo odletí, ale říká, kam letadlo udělá první zatáčku.

Typ letadla

Typem letadla se myslí zařazení daného letadla do určité kategorie. Tyto kategorie budou v této práci využívány, a budou definovány dále.

2.5 Přednost letadel

Letadla čekající ve frontě na start musejí čekat vždy ze stejného důvodu, a tím jsou přistávající letadla. Letadlo v režimu letu má totiž větší spotřebu, než letadlo stojící na zemi. Je proto snahou každého letiště nechat letadlo přistát co nejdříve.

Z tohoto důvodu se startující letadla musejí odstartovat v časových oknech, která vznikají mezi přistávajícími letadly. Z hlediska efektivity je nejvýhodnější, když startující letadlo přijede ze stojánky na ranvej právě ve chvíli, kdy začíná dané časové okno a nikde nebude zbytečně čekat. Zároveň je občas vhodné některé

přistávající letadlo navádět na přistání tak, aby časová okna byla z pohledu řízení letového provozu ideální, tedy nebyla zbytečně krátká nebo dlouhá.

Příklad

Odlétající letadlo potřebuje pro svůj start 60 sekund. Pokud by přistávající letadlo začalo provádět přiblížení k letišti v určitém bodě, tak by přistálo za 50 sekund. V tomto případě by muselo startující letadlo počkat nejméně 50 sekund. Je proto vhodnější změnit bod začátku přiblížení tak, aby přistávající letadlo přistálo za 60 sekund.
V tomto případě startující letadlo stihne odstartovat.

2.6 Dělení letadel

Letadla jsou z důvodů navigace dělena do tří kategorií:

Light - lehká letadla do hmotnosti 7 000 kg. Silueta používaná v této práci je zobrazena na obrázku 2.2a.

Medium - střední letadla od hmotnosti 7 000 kg do hmotnosti 136 000 kg. Je to většina letadel typu Airbus, Boeing, ATR a na pražském letišti představují zhruba 95% provozu. Silueta používaná v této práci je zobrazena na obrázku 2.2b.

Heavy - těžká letadla nad 136 000 kg. Např. Airbus A310, A330, A340, Boeing B767, B747, B777. Silueta používaná v této práci je zobrazena na obrázku 2.2c.

Toto rozdělení je z důvodu Wake turbulence[3], česky řečeno úplavu. Tabulka 2.1 popisuje v sekundách minimální rozestup mezi těmito třemi kategoriemi pro přistávající letadla[4], a tabulka 2.2 pro odlétající letadla[5]. Tyto časové hodnoty se dají velmi dobře odpozorovat z reálného provozu, případně mohou být tyto tabulky měněny podle aktuálních podmínek. Jako důvod těchto změn lze uvést například mlhu. Za zhoršených povětrnostních podmínek se totiž tyto rozestupy zvětšují skoro dvojnásobně.

Dále mohou existovat různé časové rozestupy pro lety letadel do různých směrů. Algoritmy popisované dále v této práci tyto rozestupy nepoužívají (používají jen výše uvedené dělení letadel), ale z návrhů algoritmů čtenář jistě pochopí jakým způsobem by se toto další dělení v algoritmech projevilo.

[3]Turbulence v úplavu se tvoří za letícím letadlem ve vzduchu. Tyto turbulence mohou být nebezpečné pro následující letadla. Proto musí být udržována minimální vzdálenost mezi letadly. Přesnější informace o úplavu lze najít v[5].

[4]definováno v[3] část 5.8.2.

[5]definováno v[3] část 5.8.3.

Typ minulé letadlo	Typ aktuální letadlo		
	Light	Medium	Heavy
Light	60	60	60
Medium	180	60	60
Heavy	180	120	60

Tabulka 2.1: Minimální vzdálenosti letadel na ranveji z pohledu turbulence v úplavu pro přistávající letadla

Typ minulé letadlo	Typ aktuální letadlo		
	Light	Medium	Heavy
Light	60	60	60
Medium	120	60	60
Heavy	180	180	60

Tabulka 2.2: Minimální vzdálenosti letadel na ranveji z pohledu turbulence v úplavu pro odlétající letadla

Příklad úplavové optimalizace

Z tabulky 2.2 lze vyčíst, že letadlo z kategorie Heavy bude muset čekat méně než z kategorie Light. V této práci je na tuto tabulku brán velmi velký zřetel, protože se v ní nachází velká možnost úspory času potřebného pro odlety letadel. Zde jsou uvedeny dvě situace pořadí startu letadel.

Pořadí Light, Medium, Heavy zabere dle tabulky 2.2 celkem 120 sekund,

Pořadí Heavy, Medium, Light zabere dle tabulky 2.2 celkem 300 sekund,

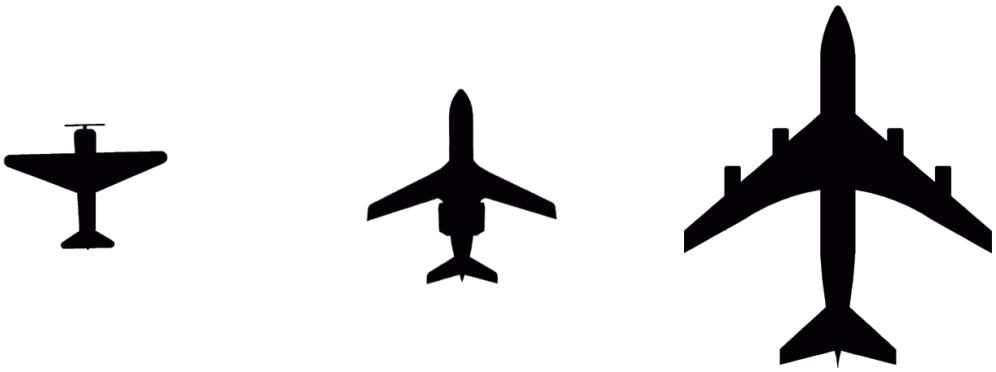
Z těchto dvou situací je efektivnější první situace, jelikož oproti druhé vznikla úspora 180 sekund. V těchto 180 sekundách by mohlo vystartovat ještě několik dalších letadel.

Poznámka

Tato tabulka se pravděpodobně v brzké době rozroste o novou základní kategorii. Tato kategorie je dána novým letadlem Airbus A-380. Ale pro naši práci budeme uvažovat zatím jen tyto tři kategorie.

2.7 Omezení provozu na dané ranveji

Kategorie letadel definované v části *2.6 Dělení letadel* jsou důležité pro povolení startů pro různé ranveje. Ranvej může mít definovaná omezení pro starty určitých letadel v určitou dobu. Toto rozdělení muže být velice podrobné, protože se v něm zohledňuje mnoho okolností provozu. Například typ motoru, které má



(a) Light (b) Medium (c) Heavy

Obrázek 2.2: Obrázky letadel

dané letadlo(z důvodů hlučnosti), atd.
Tato omezení lze prezentovat tabulkou 2.3.

Typ letadla	Čas v sekundách od počátku dne (od / do)					
	0 21599	21600 35999	36000 57599	57600 68399	68400 86399	79200 86399
Light	1	2	1	2	1	1
Medium	K	1	1	1	1	K
Heavy	K	1	2	1	2	K

Tabulka 2.3: Tabulka preference jedné ranveje pro daný čas a typy letadel

Číselné hodnoty v této tabulce říkají, jaká je preference dané ranveje pro dané letadlo a daný čas a písmeno K značí, že se v daném čase daná ranvej pro tento typ letadla nepoužívá. K bude následně v provozu definováno velmi velkým číslem. Např. 100. Algoritmus musí být schopný se vypořádat i s provozními nepřesnostmi. Tedy např., že letadlo, které již nemá danou ranvej používat, bude odlétat se zpožděním. Pokud by ještě existovala možnost odletu zpožděného letadla, tak aby splnilo požadavek na omezení na dané ranveji, tak se algoritmus bude snažit letadlo posunout do této doby. Ale v případě, že tato možnost již nemůže být využita, tak K ve všech řešeních pozmění výsledek stejným způsobem. Toto letadlo tedy nebude mít na přidělování času odletu ostatních letadel žádný vliv.

Popis hodnot v tabulce

Tabulka 2.3 tedy říká, že letadla kategorie Medium a Heavy nemohou odlétat v noci. Letadla kategorie Light by neměla odlétat v době provozní špičky, která je v časech 6:00–10:00 a 16:00–19:00. Důvodem může být například to, že tato letadla potřebují větší rozestup před svým startem, což není ve špičce, která je v těchto časech, vhodné. Letadla typu Heavy by neměla odlétat mimo špičku. Jako důvod může být uvedena jejich hlučnost.

Poznámka

Zároveň tabulka 2.3 může existovat nejenom takto obecně, ale také může existovat ve čtyřech podverzích, které určují, do kterého směru letadlo startuje. Jako příklad by se dalo uvést, že v Praze při používání ranveje 24 jsou vrtulová letadla, která startují na jih a východ posílána startovat od poloviny dráhy 13.

2.8 Odletový slot

První význam

Mezi problémy, které musí tato práce také řešit, patří odletový slot. Odletový slot je časový interval $[t-5\ minut, t+10minut]$, ve kterém má letadlo odletět. Pokud se mu to nepovede, tak se musí žádat mezinárodní organizace EUROCONTROL[6], která na základě znalostí letových plánů ostatních letadel ve vzdušném prostoru celé Evropy, přiřadí jiný časový slot. Bohužel nově přidělený slot může být přidělen se značným časovým odstupem, protože například na cílovém letišti by v čase příletu nebyl dostatek volných slotů na přistání, případně by byl velký provoz na koridoru. Účelem tedy je, aby letadlo odletělo v daném časovém intervalu. Tyto odletové sloty budou zmiňovány ve všech kapitolách této práce kromě kapitoly *7 Globální pohled na úlohu dynamického programováni*.

Druhý význam

V kapitole 7 bude odletový slot popisován v druhém svém významu. Je jím časový interval, pro který je určeno, kolik letadel v něm může maximálně odletět. Tyto sloty jsou přidělovány pravidelným letadlům na celou sezónu. Nepravidelnému letadlu je slot přiřazen tak, aby nebyl překročen maximální počet letadel v daném slotu. Toto přidělování se děje z důvodu, aby nechtělo v určitou dobu odlétat více letadel než je přípustné. Sloty jsou přidělovány organizací Slot Coordination Prague[7]. U této organizace lze také najít aktuální rozdělení slotů. Sloty nejsou jen odletové, ale i příletové. V této práci se zabýváme starty letadel, proto jsou pro nás zajímavé jen odletové sloty.

2.9 Přechod mezi APP a ACC

Posledním velkým problémem, který se musí často řešit ohledně startujících letadel, je jejich vzdálenost při odletu z oblasti letiště. Pro objasnění tohoto problému musíme vysvětlit, jakým způsobem jsou letadla navigována v globálním měřítku. Celé navigování letadel je rozděleno do tří kategorií: TWR, APP, ACC.
Problém vzniká při předávání letadla z APP do ACC, kdy v APP musí mít letadla rozestup alespoň 3 míle. V ACC musí mít rozestup alespoň 5 mil. Rozestup 3 míle je dodržován i v TWR, takže není problém s přechodem mezi TWR a APP. Zároveň musí letadla létat po určených koridorech, čímž vzniká další

[6]European Organisation for the Safety of Air Navigation
[7]Webová stránka `http://www.slot-czech.cz`

problém. Pokud by dvě letadla po startu měla letět stejným směrem, tak při přechodu mezi APP a ACC nebudou v požadovaném rozestupu. Tento problém muže být vyřešen v APP za pomocí tzv. vektorování letadel. Bohužel toto vektorování je velmi náročné. Proto je lepší těmto dvojicím letadel letícím stejným směrem předcházet. Případně mezi nimi vytvářet při startu větší než minimální rozestup. Bohužel TWR je tlačeno k tomu, aby startující letadla odlétala z letiště co nejdříve, takže možnost vytváření větších rozestupů mezi startujícími letadly je velmi malá. V tomto ohledu je lepší mezi tato letadla vložit jiné, které letí jiným směrem, čímž se mezi námi zvažovanými letadly vytvoří 6-ti mílový rozestup. Definice rozestupu pro vzlety letadel letících stejným směrem jsou definovány v[3] kapitola 5.6.2. Tento rozestup je ve většině případů 120 sekund.

2.10 Odlety do směru proti směru ranveje

Pro vytvoření úlohy dynamického programování musíme znát ještě jednu důležitou informaci. Je to informace o tom, kterým směrem jsou po startu směřována letadla, která odlétají ve směru proti směru ranveje. Důvodem je, že letadla mohou z bezpečnostních důvodů odlétat po startu z ranveje jen do tří směrů a do směru proti směru startu se otáčejí až v oblasti navigované APP. Ve všech příkladech v této práci předpokládáme, že je používána ranvej 24[8], směřujeme letadla letící na východ nejdříve směrem na sever[9]. Tato informace se projevuje jako Směr odletu, která je definovaná v podkapitole *2.4 Definice popisu letadel*

2.11 Dělení letadel dle různých typů úloh

Pro různé typy úloh, které se v této práci vyskytují je vhodné rozdělit letadla, kromě dělení v části *2.6 Dělení letadel*, dle dalšího klíče a to následujícím způsobem.
Letadla na letišti, která jsou relevantní pro tuto práci (již zaslala na TWR informaci, že jsou Ready[10]) lze rozdělit do pěti kategorii:

1. **Letadlo na ranveji.** Na dané ranveji může být v v jednom okamžiku aktivní jen jedno letadlo.

2. **Letadla již stojící v řadě před ranvejí.** Letadla stojící v řadě před ranvejí již nelze mezi sebou přesouvat. Stojí v řadě jedno za druhým a čekají na start.

3. **Letadla stojící na své stojánce.** Letadla, stojící na stojánce. Tato letadla jsou v této diplomové práci zmiňována nejvíce.

4. **Letadla mezi stojánkou a řadou před ranvejí.** Letadla na cestě mezi stojánkou a řadou před ranvejí. Těmito letadly se tato diplomová práce bude zaobírat jen v určitých kapitolách.

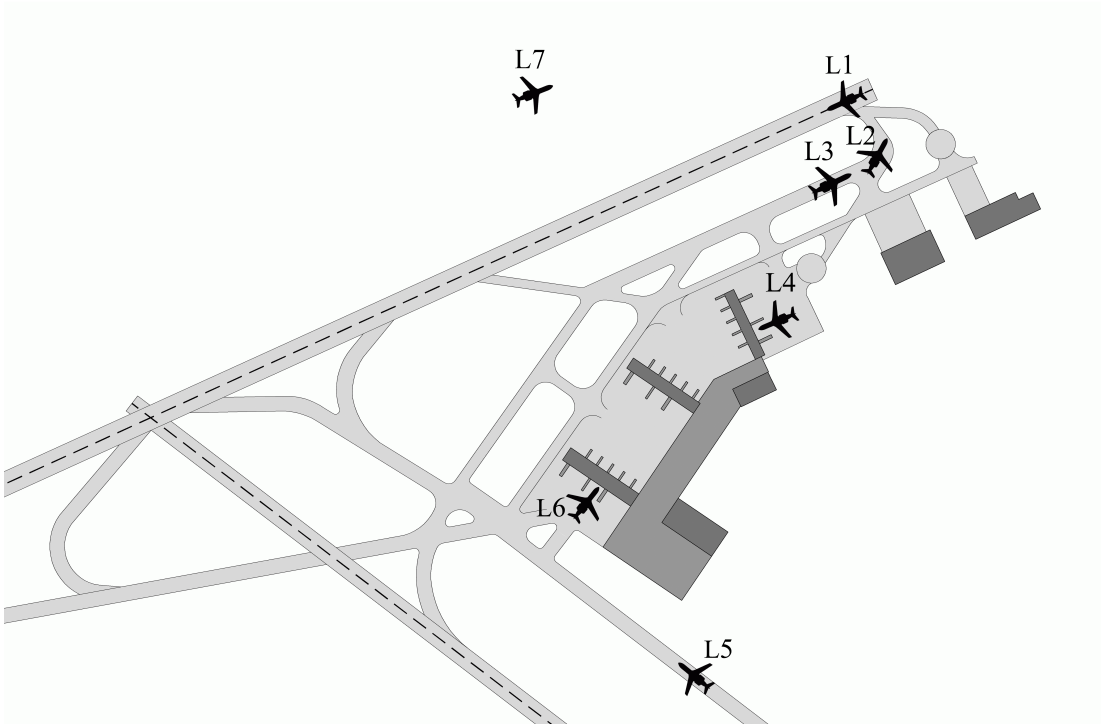
[8]Nejčastěji používaný směr v Praze
[9]standardní postup směřování letadel v Praze při použití ranveje 24
[10]letadlo dává věži na vědomí, že je připraveno startovat motory a být vytlačováno

5. **Přistávající letadla.** Letadla, která jsou naváděna na přistání.

Důležité je uvědomit si, že pořadí v řadě před ranvejí není stejné jako fronta pro danou ranvej. Zároveň si musíme uvědomit, co letadla v řadě znamenají pro letiště. **Stojí s nastartovanými motory**. Je tedy zřejmé, že účelem je, aby tato řada byla co nejkratší, případně prázdná.

Příklad

Na obrázku 2.3 je vyobrazena fronta letadel označených $L_1, L_2, L_3, L_4, L_5, L_6, L_7$, v aktuálním čase T.



Obrázek 2.3: Letiště s letadly

L_1 se rozjíždí na start. Má startovat v čase T.

L_2 čeká v řadě jako první letadlo. Má startovat v čase T + 60 sekund.

L_3 čeká na své stojánce, odkud mu cesta do řady trvá 90 sekund. Má startovat v čase T + 120 sekund.

L_4 pojíždí ze stojánky a do řady mu to bude trvat ještě 150 sekund. Má startovat v čase T + 180 sekund.

L_5 čeká na své stojánce, odkud mu cesta do řady trvá 160 sekund. Má startovat v čase T + 240 sekund.

L_6 čeká na své stojánce, odkud mu cesta do řady trvá 160 sekund. Má startovat v čase T + 300 sekund.

L_7 je přistávající letadlo. Přistává v čase T + 360 sekund (přistávající letadla budou zobrazována nad ranvejí a budou směřovat proti směru ranveje).

V tomto příkladě jsou letadla

L_1 v první kategorii definované v této podkapitole,

L_2,L_3 je v druhé kategorii,

L_4,L_6 jsou třetí kategorie,

L_5 je čtvrtá kategorie,

L_7 je pátá kategorie.

2.12 Fronta / řada letadel

Pod pojmem fronta letadel se myslí pořadí, v jakém budou daná letadla ranvej využívat. Řada letadel je pořadí letadel, ve kterém letadla stojí před ranvejí. Proto fronta obsahuje celou řadu, ale řada neobsahuje frontu. Na obrázku 2.3 řada letadel obsahuje letadla L_2,L_3 a fronta letadel obsahuje letadla $L_1, L_2, L_3, L_4, L_5, L_6, L_7$.

Kapitola 3

Úloha dynamického programování

3.1 Definice diskrétního deterministického procesu

Pro definici úlohy dynamického programování je nutno nadefinovat pojem diskrétního deterministického procesu. Tento proces je definován v[1] kapitola č.1 následujícím způsobem:

Nechť je S systém, jehož stav lze v každém časovém okamžiku z předem známého časového intervalu $[t^{start}, t^{end}]$ popsat jednoznačně n–ticí reálných čísel x_α, $\alpha = 1, \ldots, n$. Budeme přitom předpokládat, že množina všech možných stavů systému S tvoří nějakou podmnožinu X n–rozměrného euklidovského prostoru E_n. To znamená, že je–li $x = \{x_1, \ldots x_n\}$ n–tice popisujících jednoznačně některý z možných stavů systému S, je $x \in X \subset E_n$. Prostor E_n se nazývá prostor stavů příslušných k systému S. Proměnné x_α, $\alpha = 1, \ldots, n$ se nazývají stavové proměnné příslušné systému S. Množina X se nazývá množinou přípustných stavů systému S.

Nechť

$$t^{start} = t^1 < t^2 < \ldots < t^i < \ldots < t^{i+1} < t^N = t^{end}; N \geq 1$$

je nějaké dělení časového intervalu $[t^{start}, t^{end}]$. Budeme předpokládat, že systém S, který se nachází v časovém okamžiku t^1 ve stavu $x^1 \in X$ se mění v intervalu $[t^{start}, t^{end}]$ následujícím způsobem:

- Stav x^i zůstává v každém časovém intervalu $[t^i, t^{i+1}), i = 1, \ldots, N-1$ nezměněn; změna stavu nastává pouze v jednotlivých diskrétních časových okamžicích $t^1, t^2, \ldots, t^N$.
- Stav x^{i+1} závisí pouze na stavu x^i a na jistém rozhodnutí u^i v čase t^i, tedy

  $$x^{i+1} = T_i(x^i, u^i), i = 1, \ldots, N,$$

  přičemž zobrazení T_i nezávisí na stavech $x^j, 1 \leq j < i$.

- Rozhodnutí u^i lze jednoznačně definovat m–ticí reálných čísel $u_1^i, \ldots, u_m^i$. Budeme přitom předpokládat, že $u^i \equiv \{u_1^i, \ldots, u_m^i\}$ patří do dané množiny $U \subset E_m$

Prostor E_m nazýváme prostorem rozhodnutí příslušným k systému S. Proměnné $u_\beta, \beta = (1, \ldots, m)$ se nazývají rozhodovací proměnné příslušné k systému S. Množina U se nazývá množina možných rozhodnutí příslušná k systému S.

V časové intervalu $[t^{start}, t^{end}]$ se pak odehrává jistý rozhodovací proces se zadaným počátečním stavem, v němž je počáteční stav charakterizován bodem x^1, a koncový stav bodem x^{N+1}, přičemž tento koncový stav závisí jen na počátečním stavu x^1 a rozhodnutích $u^1, \ldots, u^N$. Transformační zobrazení T_i vzhledem k výše nadefinovaným vlastnostem tento tvar

$$x^{i+1} = T_i(x^i, u^i), i \ = 1, \ldots, N,$$

který je systému S jednoznačně předepsán.

Každá posloupnost

$$P_N = \{u^1, \ldots, u^N\},$$

s vlastností $u^i \in U \ , i \ = 1, \ldots, N$, přičemž

$$x^{i+1} = T_i(x^i, u^i) \in X, i \ = 1, \ldots, N \ , x^1 \in X,$$

se nazývá přípustná strategie uvažovaného rozhodovacího procesu se zadaným počátečním stavem x_1.

Množinu všech přípustných strategií označme $P_N(x_1)$

3.2 Optimalizační úloha přiřazená diskrétnímu deterministickému procesu

N–stupňovému diskrétnímu deterministickému procesu definovaném v podkapitole 3.1 přiřadíme funkci

$$F_N(x^1, \ldots, x^N, u^1, \ldots, u^N).$$

tuto funkci nazýváme cílovou funkcí přiřazenou zadanému rozhodovacímu procesu. Cílová funkce musí splňovat dva předpoklady:

- Funkce F je definována pro každé přirozené číslo N, to znamená, že je dána posloupnost funkcí

  $$F_1(x^1, u^1), F_2(x^1, x^2, u^1, u^2), \ldots, F_N(x^1, \ldots, x^N, u^1, \ldots, u^N),$$

  které přísluší určitý rekurentní předpis, podle kterého jsou tyto funkce vytvářeny.

- Funkci $F_N(x^1, \ldots, x^N, u^1, \ldots, u^N), N \geq 2$ lze vyjádřit pomocí funkce $F_{N-1}(x^1, \ldots, x^{N-1}, u^1, \ldots, u^{N-1})$ a zadané funkce $G_N(x^N, u^N)$, to znamená, že bude platit

$$F_N(x^1, \ldots, x^N, u^1, \ldots, u^N) = F_{N-1}(x^1, \ldots, x^{N-1}, u^1, \ldots, u^{N-1}) + G_N(x^N, u^N)$$

Příslušná optimalizační úloha na tomto rozhodovacím procesu spočívá nyní v tom, že hledáme takovou strategii,

$$*P_N = \{*u^1, \ldots * u^N\},$$

která má vlastnost

$$F_N(x^1, *u^1, \ldots * u^N) = max_{P_N(x_1)} \left\{F_N(x^1, u^1, \ldots u^N)\right\}$$

tedy neexistuje strategie, která by dávala větší hodnotu funkce f.

Jestliže existuje přípustná strategie $\{*u^1, \ldots * u^N\}$ s touto vlastností, pak se nazývá optimální strategií dané optimalizační úlohy procesu s cílovou funkcí $f(x^1, \ldots, x^N, u^1, \ldots, u^N)$.

U většiny problému z praxe, které vedou na úlohu dynamického programování je cílová funkce separabilní.

$$F_N(x^1, \ldots, x^N, u_1, \ldots, u^N) = \sum_{i=1}^{N} G_i(x^i, u^i)$$

Zformulujme nyní na N-stupňovém disktrétním rozhodovacím procesu následující optimalizační úlohu:

Budíž

$$F_N(x^1, \ldots, x^N, u^1, \ldots, u^N) = \sum_{i=1}^{N} G_i(x^i, u^i)$$

daná separabilní cílová funkce, která je definována na množině všech bodů $x^i \in X \subset E_n$,$u^i \in U \subset E_m, i = 1, \ldots, N$. Nechť $P_N(x^1)$ je množina všech přípustných strategií.

Úkolem je určit maximum cílové funkce F na množině $P_N(x^1)$, to znamená, že je třeba nalézt takovou strategii $*P_N = \{*u^1, \ldots * u^N\}$, která má následující dvě vlastnosti:

- Neexistuje jiná strategie, která by dávala lepší hodnotu cílové funkce.
- Strategie je přípustná, tedy platí $*x^{i+1} = T_i(*x^i, *u^i) \in X, i = 1, \ldots, N$, $* x^1 = x^1$

3.3 Optimální rozdělení zdrojů

Protože v dalších částech práce budeme používat metodu dynamického programování, která se používá pro optimální rozdělení zdrojů, tak je vhodné tuto metodu představit.

Máme M zdrojů o kapacitách $a = (a_1, \ldots, a_M)$. Pro každý zdroj a_l platí, že ho lze rozdělit následujícím způsobem: $a_l^1, \ldots, a_l^q$, kde $0 = a_l^1 < a_l^2 < \ldots < a_l^q$
Existuje N zájemců o zdroje, kde $N > 1$. Ti požadují jednotlivé zdroje v množství $K_l^i \geq 0, l = 1, \ldots M, i = 1, \ldots N$

Známe funkce G_i uvádějící výnos zájemců (i = 1, ..., N) na základě přiřazených zdrojů (l = 1, ..., M).

- Rozhodovací proměnné : $u^i = (u_1^i, \ldots, u_M^i)$ udává množství zdrojů přiřazených i–tému zájemci.
- Stavové proměnné : $x^i = (x_1^i, \ldots, x_n^i)$ udává množství zbývajících zdrojů, které je ještě možné dělit.
- Stavová transformace : $x^{i+1} = x^i - u^i, \forall i, x^1 = a$.

Dále definujeme
$U = \{u^i | u_l^i \in \{a_l^1, \ldots, a_l^q\}, i = 1, \ldots, N, l = 1, \ldots, M\}$,
$X = \{x^i | x^i \in [0, a]\}$ (ale ne spojitě, ale jen $a^1 < a^2 < \ldots < a^q$),
Následně bude úloha pro hledání optimálního rozdělení zdrojů bude hledat tuto strategii v množině:
$P_N(x^1) = \left\{(u^1, \ldots, u^N) | u^i \in U, u^i \leq x^i \& u^i \leq K^i, i = 1, \ldots, N, x^1 = a\right\}$,
Optimální strategii při cílové funkci
$max_{P_N(x^1)} \sum_{i=1}^{N} g_i(u^i)$ kde $g_i(u^i) \geq 0$.

Kapitola 4

Nejjednodušší úloha optimálního rozdělení zdrojů pro letiště

Nejjednodušší úloha optimálního rozdělení zdrojů, kterou budeme řešit v sobě obsahuje jen určitá omezení.

Letiště má jednu ranvej určenou jen pro starty a do optimalizační úlohy se zahrnují jen letadla, která stojí na stojánkách. Nejsou zde zahrnuty požadavky na sloty letadel. V tomto případě jde v optimalizační úloze jen o splnění požadavků na rozestupy letadel a zařazení požadavku, aby dvě letadla neletěla po startu stejným směrem. Za těchto omezení se v této úloze vyskytují z pohledu dělení letadel v části *2.11 Dělení letadel dle různých typů úloh* jen následující letadla:

Všechna letadla stojící stojánce. Jejich počet se označuje proměná h. Na obrázku 4.1 jsou to letadla L_1, L_2

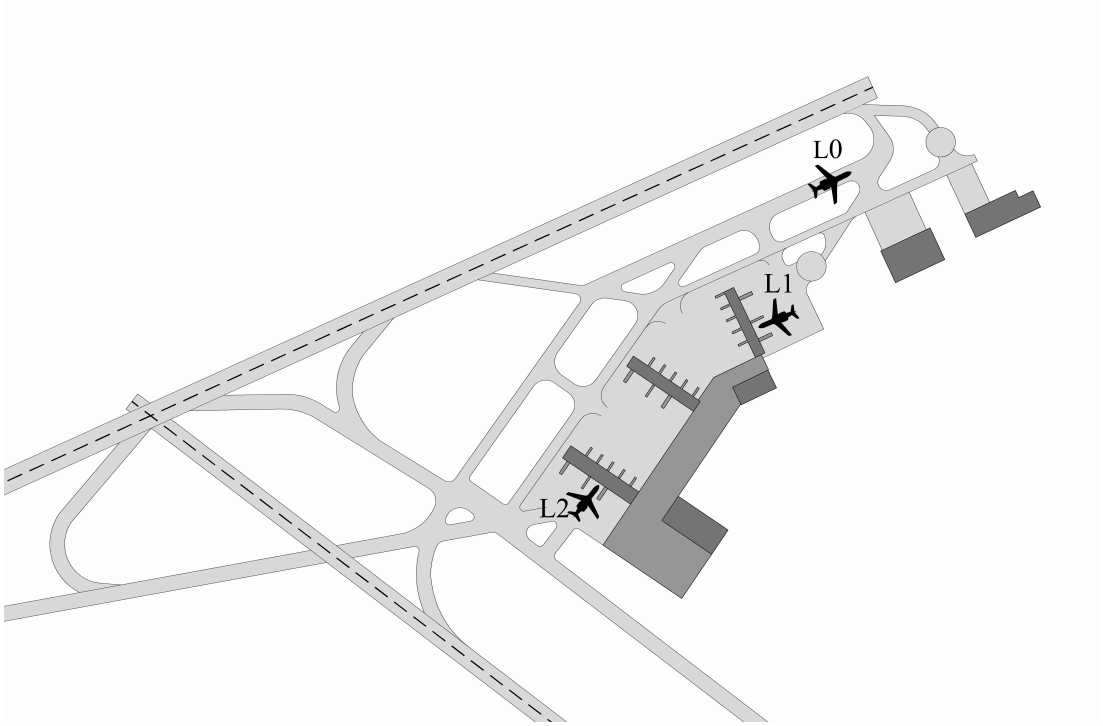
Poslední letadlo v řadě před ranveji (případně může být ranvej volná). Na obrázku 4.1 je to letadlo L_0

Na obrázku 4.1 jsou z obrázku 2.3 zobrazena jen ta letadla, která jsou v této nejjednodušší úloze dynamického programování použita. Zároveň bylo provedeno jejich přečíslování tak, aby odpovídalo číslování v této úloze.

Pro jednodušší představu se dá celý problém popsat následujícím způsobem. Po určitou dobu se sbírají informace z letadel, která stojí na stojánce a podala rádiem oznámení, že jsou Ready. Následně se řeší úloha dynamického programování, která je vytvořena těmito letadly. Čas pojíždění letadla ze stojánky na ranvej je v této úloze uvažován. Letadla, která jsou již na cestě ze stojánky na ranvej, nebudou v této úloze už zvažována. Jako čas t_0 označíme čas, ve kterém má startovat poslední naplánované letadlo (které označíme L_0). Toto letadlo L_0 je poslední letadlo ve frontě letadel čekajících na ranvej.

4.1 Popis úlohy optimálního rozdělení zdrojů

V podkapitole 3.3 jsme zformulovali úlohu pro optimální rozdělení zdrojů. Úloha musí obsahovat zdroje, konzumenty, stavové proměnné, rozhodovací proměnné,



Obrázek 4.1: Obrázek letiště obsahující letadla, která se budou v této úloze používat

stavovou transformaci. Abychom ukázali její souvislost z naším problémem, je nutno definovat zdroje a konzumenty. Z logického hlediska je zdrojem ravej a konzumentem jsou letadla. Přesněji zdrojem není samotná ranvej, ale zdrojem je čas, po který letadlo nechává ranvej po sobě nepoužitelnou pro další letadla. Z pohledu úlohy se tedy bude jednat o rozdělení času mezi letadla. Přesněji se bude jednat o časové intervaly, které budou přiřazeny letadlům, přičemž dva intervaly mohou mít jen jeden společný bod. Tímto bodem jsou krajní body obou intervalů.

Zdroje budou v naší úloze označovat určité časové kvantum. Pokud bude letadlu tento zdroj přidělen, tak to znamená, že letadlo může toto časové kvantum využívat danou ranvej. Každý zdroj má navíc určený časový okamžik dne, kterému je přiřazen. Takže letadlu je při přidělení zdroje řečeno, že může danou ranvej využívat určité kvantum času od určitého časového okamžiku dne. Stavová proměnná bude udávat, které zdroje je možno v daném stavu použít. Rozhodovací proměnná bude určovat, které zdroje mají být přiřazeny danému letadlu. Transformační proměnné budou určovat, které změny stavů lze při určitých rozhodnutích provádět.

4.2 Matematická formulace úlohy optimálního rozdělení zdrojů

Nechť máme úlohu o N letadlech. Pak můžeme následovně definovat

Zdroje

$$a = \{a_1, a_2, \ldots, a_{3 \cdot N}\}.$$

Stavové proměnné

$$X = \{(1,1,1,1,\ldots,1,1),(1,1,1,1,\ldots,1,0),\ldots,(0,0,0,0,\ldots,0,0)\},$$

kde $\forall x \in X$ je mohutnost$(x) = 3 \cdot N$.

$$x^1 = (1,1,1,1,\ldots,1,1).$$

Rozhodovací proměnné

$$\begin{aligned} U \quad = \{ \quad & (1,0,0,0,\ldots,0,0,0),(0,1,0,0,\ldots,0,0,0),\ldots,(0,0,0,0,\ldots,0,0,1), \\ & (1,1,0,0,\ldots,0,0,0),(0,1,1,0,\ldots,0,0,0),\ldots,(0,0,0,0,\ldots,0,1,1), \\ & (1,1,1,0,\ldots,0,0,0),(0,1,1,1,\ldots,0,0,0),\ldots,(0,0,0,0,\ldots,1,1,1)\}, \end{aligned}$$

kde $\forall u \in U$ je mohutnost$(u) = 3 \cdot N$.

Transformační funkci

$$x^{i+1} = T_i(x^i, u^i) = x^i - u^i, \forall i$$

Množinu přípustných strategií

$$P_N(x^1) = \{(u^1, \ldots u^N) | u^i \in U, \forall i, x^{i+1} = T_i(x^i, u^i) \in X, x^1 \in X\}$$

Cílovou funkci

$$max_{P_N(x^1)} \sum_{i=1}^{N} G_i(x^i, u^i)$$

přičemž platí

$$G_i(x^i, u^i) = \frac{(p_i - r_i)}{\alpha^i}$$

kde r^i je počet zdrojů přidělený daným rozhodnutím u^i, p_i je maximální počet nutných zdrojů pro letadlo L_i a α^i je preference ranveje pro dané letadlo.

Všechny výše uvedené definice budou v následujících kapitolách upravovány takovým způsobem, aby úloha optimálního rozdělení zdrojů reálnému provozu na letišti.

4.3 Velikost základní časové jednotky

Základní časová jednotka je takový časový interval, který je největší společný dělitel všech intervalů, které musí všechna letadla dodržet v možných závislostech na možných předchozích letadlech. Pokud se podíváme do tabulek 2.1 a 2.2, tak z nich vyčteme, že nejmenší společný dělitel je minimem z těchto hodnot a je jím 60 sekund.

4.3.1 Zdroj

V podkapitole 3.3 jsou zdroje označovány písmenem a. Pro každý zdroj je definováno jeho dělení.

Nechť můžeme jednu ranvej používat právě jednu hodinu, což je 3600 sekund. Toto kvantum času bude přiřazeno jednomu zdroji. Protože bude dělení odpovídat základní časové jednotce, tak bude zdroj rozdělen na 60 stejných částí. Tedy bude platit že $0 < q$ a $q \leq 60$, přičemž q je jako počet dělení zdroje.

Tato definice zdroje má jednu nevýhodou. Tou je, že úloha rozdělení zdrojů je schopná pro více ranvejí najít optimální rozdělení letadel na ranveje, ale neřekne v jakém pořadí mají daná letadla na dané ranveji startovat.

Proto je vhodnější nevytvářet jeden zdroj pro jednu ranvej. Ale vytvořit více zdrojů pro jednu ranvej, které budou na sebe časově navazovat. Tedy místo jednoho zdroje o velikosti 60 jednotek, vytvoříme 60 zdrojů o velikosti jedné jednotky (zdroje nebudou mit v sobě žádné dělení, vždy bude muset být využít celý zdroj), přičemž v úloze musí platit, že pokud budou daným spotřebitelem použity zdroje $a^i, a^j, i < j$, tak musí být spotřebitelem úplně použity všechny zdroje a^k, takové že platí $i < k < j$.

Pro více ranvejí by byly tyto zdroje určeny tak, aby bylo zřejmé, že letadlo může využívat jen zdroje jedné ranveje. Což bude zajišťovat rozhodnutí úlohy.

Pro každý zdroj a_i definujeme hodnotu t_i jako čas jeho počátku, který je zdroji jednoznačně přiřazen. V této části práce jsme definovali, že zdroje obsahují nějaký počet základních časových jednotek. Protože časová jednotka obsahuje určité kvantum času, tak i zdroj obsahuje určité kvantum času. Proto si lze zdroje představit jako určité časové intervaly. Bylo řečeno, že zdroje na sebe navazují, tedy že tyto časové intervaly na sebe v krajních bodech navazují. Pokud tyto časové intervaly přiřadíme k určitým intervalů času v průběhu dne, tak můžeme určit, kdy daný časový interval začíná. tento časový údaj označuje hodnota t_i. Protože jednotka obsahuje určitý časová interval a jednotky na sebe navazují, tak lze prohlásit, že rozdíl časů t_i a t_{i+1} je roven velikosti časového kvanta které obsahuje zdroj.

Úmluva o základní časové jednotce

V této práci vždy bude velikost zdroje 60 sekund, a základní časová jednotka bude mít také velikost 60 sekund. V dalších částech práce budeme tyto hodnoty používat automaticky, aniž bychom je znova definovali.

4.4 Lidský faktor

Čtenáře by mohlo napadnout, že letadla nikdy nestartují přesně podle údajů z počítače, ale existuje nějaký lidský faktor, který je může zdržet. Například tak, že pilot ještě nestihl udělat celou předletovou přípravu. Zároveň se všechny příkazy pro start letadla předávají pracovníkem řídícího letového provozu pilotům s nějakým časovým předstihem, díky čemuž se podaří, že občas letadla odstartují dříve než by měla. Jak zpoždění, tak tento předstih jsou v průměru všech letadel

v daném dnu, v poměru k 60 sekundám tak zanedbatelné, že tento lidský faktor není nutno v této úloze brát v úvahu. Proto se budeme zaobírat jen přesným teoretickým řešením, s tím, že případné malé praktické odchylky se navzájem vyrovnají.

4.5 Cíl úlohy

Úlohy definované v kapitole *3 Úloha dynamického programování* se zaobírají tím, že se úloha snaží maximalizovat zisk s omezeným počtem zdrojů. V tomto případě bude úloha jiná. Zde máme teoreticky zdrojů nekonečně mnoho, ale naším úkolem je minimalizovat dobu, po kterou je ranvej letadly obsazena, tedy minimalizovat počet použitých zdrojů. Jinak řečeno, letadlo musí odletět, z pohledu navigace letadel je vhodné, aby letadla odletěla co nejdříve, a zabralo jim to co nejméně času.

4.6 Projekce času

V úvodních kapitolách této práce jsou uváděna různá omezení, která jsou vztahována k časovému okamžiku dne. V definici zdrojů jsme definovali t_i jako čas dne, ke kterému je přiřazen zdroj a_i. Naše úloha se zaobírá určitým časovým intervalem dne. Je proto vhodné přiřadit první zdroj a^1 aktuálnímu času, ve kterém se úloha začíná počítat a k tomuto času přepočítávat ostatní časové údaje. Tedy například a^{10} bude při dělení po 60 sekundách znamenat okamžik, který nastane za 10 minut.

4.7 Stavy systému S

V podkapitole 3.3 jsou popsány stavové proměnné. Byly označeny písmenem x. V této kapitole bude stavová proměnná popisovat, kolik ještě zbývá časových jednotek v jednotlivých zdrojích a pro jaký typ letadla a směr jeho odletu byly použité časové jednotky daného zdroje použity. Maximální počet dvojic (typ letadla, směr odletu) je pro tři typy letadel a tři směry odletu číslo devět.

Daný stav bude vždy jednoznačně určovat jaké základní časové jednotky byly již úlohou využity, a které je možno ještě využít.

Na počátku úlohy bude platit, že počáteční stav x^1 se bude rovnat a. Kde a bylo definováno na počátku podkapitoly 3.3 jako M zdrojů o určitých kapacitách.

Důležité na informaci o typu letadla a směru jeho odletu pro které byly dané časové jednotky využity je, že tato informace je důležitá pro přidělování následujícího zdroje po tomto zdroji. Například by dané dva typy letadel nemohly odlétat za sebou. Poté co takové rozhodnutí padne, tak již určitě další takové rozhodnutí se nebude daného zdroje týkat.

4.7.1 Maximální počet zdrojů pro danou úlohu

Maximální počet zdrojů určuje, jak nejdéle může námi počítaná úloha trvat.

Každý zdroj určuje určitý časový interval, který může být na dané ranveji přidělen spotřebiteli, tedy letadlu. V podkapitole 4.3 jsme psali že zdroje na sebe navazují. Tedy pokud jsou letadlu přiděleny dva na sebe navazující zdroje (např. a_i, a_{i+1}), tak mu je přidělen určitý časový interval.

V pohledu na využití zdrojů můžeme vybrat ze dvou možných popisů problému. Buď lze počítat dobu, po kterou letadlo nemůže startovat, protože před ním startovalo letadlo určitého typu, nebo lze počítat s tím, že dané letadlo odstartovalo a úloha musí vytvořit určitý časový interval, než může další letadlo odstartovat. Lépe si tyto dva přístupy lze vysvětlit následujícím způsobem. Letadlo má přidělen určitý časový interval. Pokud by úloha používala první možnost znamenalo by, že letadlo odstartuje na konci tohoto intervalu, protože ten interval by byl rozestup mezi ním a předchozím letadlem. Druhá možnost znamená, že letadlo startuje na počátku intervalu. Tento interval znamená dobu, po kterou za ním nemůže odstartovat další letadlo.

Maximální počet zdrojů pro danou úlohu, určuje jak by byla úloha velká, kdyby neproběhla žádná optimalizace a letadla by za sebou zanechávala svůj maximální rozestup.

Každý typ letadla má tento maximální rozestup jiný. Maximální časový interval, který jedno letadlo potřebuje je určován z tabulek 2.1 a 2.2.

Pro letadla typu Heavy jsou to celkem 3 základní časové jednotky (protože maximální úplav letadla je 180 sekund, a základní časová jednotka má velikost 60 sekund). Pro letadla typu Medium jsou to 2 základní časové jednotky a pro letadla typu Light by to dle tabulky o úplavu měla být jedna základní časová jednotka. Pro letadla není úplav jediným omezením. Další nutné omezení, které zvětšuje maximální počet nutných základních časových jednotek je definován v podkapitole *2.9 Přechod mezi APP a ACC*. Je jím omezení na směr odlétajících letadel. Toto omezení říká, že počet základních časových jednotek musí pokrýt nejméně 120 sekund, což jsou v našem případě dvě základní časové jednotky. Toto omezení proto ovlivní jen letadla typu Light, protože ostatní mají již přiděleny nejméně dvě základní časové jednotky.

Ve zbytku práce bude maximální počet potřebných základních časových jednotek pro dané letadlo označovat proměnnou Q. Tedy Q_i znamená maximální počet potřebných základních časových jednotek pro letadlo L_i. Zároveň nadefinujme maximální potřebný počet zdrojů pro dané letadlo jako p. Tedy p_i tedy znamená maximální potřebný počet zdrojů pro letadlo L_i. V této práci se velikost základní časové jednotky rovná velikosti zdroje, proto bude platit $Q_i = p_i$.

Tabulka *4.1 Maximální počet zdrojů pro daný typ letadla* shrnuje výše uvedené počty zdrojů a základních časových jednotek pro různé typy letadel.

definujme maximální počet potřebných zdrojů P pro úlohu L takto:

$$P = \sum_{i=0}^{h} p_i,$$

kde proměnná h byla definována společně s L v podkapitole 2.4.

Typ letadla	Q	p
Light	2	2
Medium	2	2
Heavy	3	3

Tabulka 4.1: Maximální počet zdrojů pro daný typ letadla

Příklad

Mějme úlohu L obsahující letadla $L_1, L_2, L_3, L_4, L_5, L_6$. Přičemž L_1, L_2 jsou typu Heavy, L_3, L_4 jsou typu Medium a L_5, L_6 jsou typu Light.

Maximální potřebný počet zdrojů P tedy dle tabulky 4.1 bude $P = p_1 + p_2 + p_3 + p_4 + p_5 + p_6 = 3 + 3 + 2 + 2 + 2 + 2 = 14$

Úloha tedy maximálně využije 14 zdrojů.

Výše popsaná definice maximálního počtu zdrojů předpokládá jeden nesplnitelný předpoklad. Tím je, že se letadlo dostaví na start v přiřazený okamžik, který mu byl přidělen úlohou. V praxi to tak bohužel nefunguje, protože letadlům trvá nějakou dobu cesta ze stojánky k ranveji. Tuto dobu jsme definovali v podkapitole *2.4 Definice popisu letadel* kde bylo určeno, že délka tohoto pojíždění se vyjadřuje jako $(tc)_i$. Tato hodnota byla v podkapitole 2.4, definována tak, že se jedná o počet základních časových jednotek. Proto můžeme každému letadlu určit, které zdroje z úlohy nebude moci využít, protože základní časové jednotky jsou v celé práci stejně velké. Z tohoto důvodu se musí zvětšit počet zdrojů, aby všechna letadla mohla být přiřazena nějakému zdroji.

Proto k hodnotě P získané výše uvedeným vztahem musíme přičíst nejvyšší hodnotu z hodnot $(tc)_i$, tj.:

$$P := P \ + \max_{1 \leq i \ \leq h} tc_i.$$

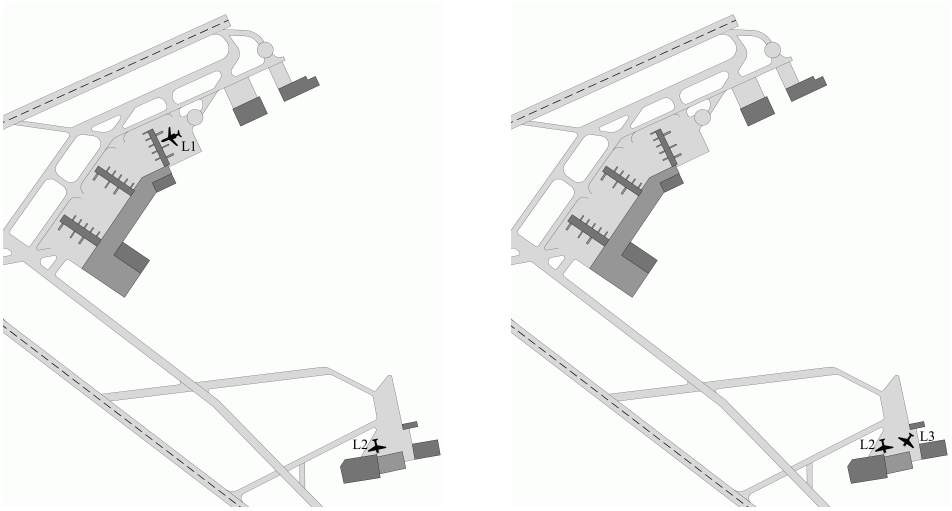
Příklad

Mějme letadla L_1, L_2, L_3 zobrazená na obrázcích 4.2 s těmito hodnotami proměnných:

$$p_1 = 3, p_2 = 3, p_3 = 3, (tc)_1 = 2, (tc)_2 = 7, (tc)_3 = 7$$

Pokud by platilo jen $P = \sum_{i=0}^{h} p_i$, v příkladu z obrázku 4.2a by letadlo L_2 nestihlo dojet k ranveji tak, aby mohlo využít jakéhokoliv zdroje. Počet zdrojů by byl 6, přičemž letadlo L_2 by mohlo začít používat zdroj, který by měl hypoteticky číslo 8. V příkladu z obrázku 4.2b by obě letadla nestihla dojet k ranveji. Přičemž důvody jsou stejné jako u prvního příkladu. V obou případech je proto nutno k P přičíst hodnotu $\max_{1 \leq i \ \leq h} tc_i$, čímž je zajištěno, že všechna letadla přijedou k ranveji tak, aby mohla využít pro sebe potřebný počet zdrojů.

Příklad na obrázku 4.2b ukazuje jeden problém ohledně velikosti úlohy. Pokud budou letadla daleko od ranveje, tak bude úloha zbytečně velká. Vhodnější je nejdříve najít $t = \min_{1 \leq i \ \leq h} tc_i$ a všechny tc_i o tuto hodnotu zmenšit. Tímto úlohu



(a) Příklad A (b) Příklad B

Obrázek 4.2: Ukázka nutnosti přičtení nejvyšší hodnoty tc_i

zmenšíme a posuneme v čase o t základních časových jednotek. Čímž nebudeme vytvářet úlohu se zdroji, které nemohou být využity. Výsledná maximální délka úlohy bude

$$P = \sum_{i=0}^{h} p_i + \max_{1 \leq i \ \leq h} tc_i - \min_{1 \leq i \ \leq h} tc_i$$

Příklad

Minulý příklad tedy bude upraven následovně:

Pro úlohu z obrázku 4.2a by se $P = 11$ a $t = 2$, tedy úloha by byla zkrácena o dva zdroje a posunuta v čase o dva zdroje.

Pro úlohu z obrázku 4.2b by se $P = 6$ a $t = 7$, tedy úloha by byla zkrácena o sedm zdrojů, a posunuta v čase o sedm zdrojů.

Pro matematickou korektnost úlohy je také nutno definovat $p_{h+1} = 0$.

4.7.2 Množina X všech stavů systému S

Množina X všech stavů systému S obsahuje takové stavy x_i, které jsou všemi kombinacemi využití základních časových jednotek všemi dvojicemi (typ letadla, směr odletu), které se v úloze vyskytují.

Společné informace pro příklady

V této podkapitole budeme v příkladech používat úlohu L obsahující letadla L_1, L_2. Přičemž L_1 je typu Light startující na sever a L_2 je typu Medium startující na jih.

V této úloze se vyskytují dvě různé dvojice (typ letadla, směr odletu). Jsou to (Light,sever) a (Medium,jih). Budeme je označovat zkratkami Ls a Mj. Maximální počet zdrojů pro tuto úlohu je $P = 4$. Stav

$$x_i \;=\; (\;\; 0Ls,\;\; 1,\;\; 1,\;\; 1\;\;)$$

Značí, že pro letadlo ze skupiny Ls byla použita základní časová jednotka ze zdroje číslo 1.

Pravidla pro stavy systému

Je zřejmé že mohutnost množiny M bude velká. Existují předpoklady v této úloze, které velikost množiny rapidně zmenšují. Charakterizují je následující pravidla:

Pravidlo 1 V daném stavu nemůže být více skupin zdrojů, které byli přiřazeny dvojici (typ letadla, směr odletu), než je letadel, které patří do této skupiny.

Pravidlo 2 V daném stavu nemůže být použito více časových jednotek příslušejících dané dvojici (typ letadla, směr odletu), než je součet časových jednotek, které letadla v dané skupině mohou maximálně využít.

Pravidlo 3 V daném stavu musí posloupnost skupin zdrojů splňovat pro každé dvě následující skupiny zdrojů musí být velikost první posloupnosti tak velká, aby zabezpečila minimální rozestup mezi typem letadla přiřazeného první skupině a letadla přiřazeného druhé skupině.

ad. Pravidlo 1

V podkapitole 4.3 definujeme následující omezení. Pokud jsou daným spotřebitelem použity zdroje $a^i, a^j, i < j$, tak musí být spotřebitelem plně použity všechny zdroje a^k, takové, že platí $i < k < j$. Tyto zdroje můžeme označit pojmem skupina zdrojů. Pokud je v úloze více letadel, která jsou ve stejné množině definovanou dvojicí (typ letadla, směr odletu), tak pro jedno letadlo se může v daném stavu vyskytovat jen jedna skupina zdrojů, která je danému letadlu přiřazena. Proto platí toto pravidlo.

Příklad

Pravidlo 1 říká, že například následující stavy nemohou existovat

$$\begin{array}{rcl} x_1 & = & (\;\; 0Ls,\;\; 1,\;\;\; 0Ls,\;\; 1\;\;\;\;\;), \\ x_2 & = & (\;\; 1,\;\;\; 0Ls,\;\; 1,\;\;\; 0Ls\;\;), \\ x_3 & = & (\;\; 0Ls,\;\; 0Ls,\;\; 1,\;\;\; 0Ls\;\;). \end{array}$$

V úloze je jen jedno letadlo, které je ve skupině Ls. A v těchto stavech jsou pro tuto skupinu Ls alespoň dvě skupiny zdrojů.

---

ad. Pravidlo 2

Pokud by v daném stavu bylo použito více časových jednotek pro danou skupinu letadel, než je součet časových jednotek, které budou letadla dané skupiny maximálně potřebovat, tak není možno některé tyto jednotky přiřadit jakémukoliv letadlu z této skupiny.

Příklad

Pravidlo 2 říká že například následující stavy nemohou existovat

$$\begin{array}{rcl} x_1 & = & (\ 0Ls,\ 0Ls,\ 0Ls,\ 1\ ) \\ x_2 & = & (\ 1,\ 0Ls,\ 0Ls,\ 0Ls\ ) \\ x_3 & = & (\ 0Ls,\ 0Ls,\ 0Ls,\ 0Ls\ ) \end{array}$$

V úloze je jen jedno letadlo, které je ve skupině Ls. Toto letadlo potřebuje maximálně dvě časové jednotky. Proto mu nemůžeme přidělit těchto časových jednotek více.

Pravidlo 3

Pravidlo č. 3 by šlo jednoduše aplikovat za pomocí definice transformační funkce, ale pokud bude toto pravidlo aplikováno již zde, tak zmenší počet možných stavů systému, což je vhodnější. Toto pravidlo říká následující. Pokud v nějakém stavu úlohy bude platit následující: jedno letadlo využívá zdroje $a^i, \ldots, a^j, i < j$ a druhé letadlo využívá zdroje $a^k, \ldots, a^l, k < l$, kde $j \leq k$, a zdroje $a^j, \ldots, a^k$ nejsou využity jiným letadlem, tak zároveň musí platit, že doba mezi první použitou časovou jednotkou pro první letadlo a poslední použitou časovou jednotkou pro první letadlo nesmí být menší, než je je minimální rozestup mezi prvním a druhým letadlem.

Příklad

Stavy odebrané pravidlem číslo 3

$$\begin{array}{rcl} x_1 & = & (\ 0Mj,\ 0Ls,\ 1,\ 1\ ), \\ x_2 & = & (\ 1,\ 0Mj,\ 0Ls,\ 1\ ), \\ x_3 & = & (\ 1,\ 1,\ 0Mj,\ 0Ls\ ), \\ x_4 & = & (\ 0Mj,\ 0Ls,\ 0Ls,\ 1\ ), \\ x_5 & = & (\ 0Mj,\ 1,\ 0Ls,\ 1\ ), \\ x_6 & = & (\ 1,\ 0Mj,\ 1,\ 0Ls\ ). \end{array}$$

Stavy x_1, x_2, x_3, x_4 jsou tímto pravidlem zakázány, protože nesplňují podmínku na minimální rozestup (letadlo typu Medium musí mít před letadlem typu Light rozestup alespoň 120 sekund).

Stavy x_5, x_6 sice z logického hlediska podmínku na minimální rozestup splňují, ale pravidlo 3 bylo nastaveno záměrně tak, aby tyto stavy vynechalo, protože

v množině stavů se budou vyskytovat následující stavy, které daný provoz popisují lépe. Jsou to tyto stavy.

$$\begin{array}{rcl} x_{5\star} & = & (\ 0Mj,\ 0Mj,\ 0Ls,\ 1\ ), \\ x_{6\star} & = & (\ 1,\ 0Mj,\ 0Mj,\ 0Ls\ ). \end{array}$$

Protože ze stavů x_5, x_6 nelze přejít do stavů, ve kterých by mohly být využity zdroje číslo 1 (v x_5) a zdroj číslo 2 (v x_6) (protože z definice minimálních vzdáleností nelze přiřadit tyto zdroje žádnému letadlu), tak je vhodnější tyto stavy z úlohy odstranit (jsou zastoupeny stavy $x_{5\star}, x_{6\star}$). Zmenší se tak celkový počet stavů.

Je velmi důležité se uvědomit, který stav je počáteční, a které stavy jsou koncové. Počáteční stav je takový stav, který je definován všemi zdroji s maximální kapacitou. Koncové stavy jsou takové stavy, které v sobě obsahují zdroje využití tak, že každý spotřebitel dostal přiděleny nějaké zdroje. Zásadní výhodou této informace je, že z takových stavů se nelze dostat jakýmkoliv rozhodnutím do jiného stavu, proto nelze provést rozhodnutí, a tedy se ani nemusí počítat hodnota cílové funkce pro tento stav a jakékoliv rozhodnutí.

Příklad na vytvoření množiny X všech stavů systému S

Nechť máme letadla L_1, L_2 definovaná na počátku této podkapitoly

Následně pak s využitím pravidel 1,2,3 vytvoříme množinu X všech možných stavů. Tato množina bude obsahovat tyto stavy

$$\begin{array}{rcl} x_0 & = & (\ 1,\ 1,\ 1,\ 1\ ) \\ x_1 & = & (\ 0Ls,\ 1,\ 1,\ 1\ ) \\ x_2 & = & (\ 1,\ 0Ls,\ 1,\ 1\ ) \\ x_3 & = & (\ 1,\ 1,\ 0Ls,\ 1\ ) \\ x_4 & = & (\ 1,\ 1,\ 1,\ 0Ls\ ) \\ x_5 & = & (\ 0Ls,\ 0Ls,\ 1,\ 1\ ) \\ x_6 & = & (\ 1,\ 0Ls,\ 0Ls,\ 1\ ) \\ x_7 & = & (\ 1,\ 1,\ 0Ls,\ 0Ls\ ) \\ x_8 & = & (\ 0Mj,\ 1,\ 1,\ 1\ ) \\ x_9 & = & (\ 1,\ 0Mj,\ 1,\ 1\ ) \\ x_{10} & = & (\ 1,\ 1,\ 0Mj,\ 1\ ) \\ x_{11} & = & (\ 1,\ 1,\ 1,\ 0Mj\ ) \\ x_{12} & = & (\ 0Mj,\ 0Mj,\ 1,\ 1\ ) \\ x_{13} & = & (\ 1,\ 0Mj,\ 0Mj,\ 1\ ) \\ x_{14} & = & (\ 1,\ 1,\ 0Mj,\ 0Mj\ ) \\ x_{15} & = & (\ 0Ls,\ 0Mj,\ 1,\ 1\ ) \\ x_{16} & = & (\ 0Ls,\ 1,\ 0Mj,\ 1\ ) \\ x_{17} & = & (\ 0Ls,\ 1,\ 1,\ 0Mj\ ) \\ x_{18} & = & (\ 0Ls,\ 0Mj,\ 0Mj,\ 1\ ) \\ x_{19} & = & (\ 0Ls,\ 1,\ 0Mj,\ 0Mj\ ) \\ x_{20} & = & (\ 1,\ 0Ls,\ 0Mj,\ 1\ ) \\ x_{21} & = & (\ 1,\ 0Ls,\ 1,\ 0Mj\ ) \end{array}$$

$$
\begin{array}{lcl}
x_{22} & = & (\ 1, \quad 0Ls, \quad 0Mj, \quad 0Mj\ ) \\
x_{23} & = & (\ 1, \quad 1, \quad 0Ls, \quad 0Mj\ ) \\
x_{24} & = & (\ 0Mj, \quad 0Mj, \quad 0Ls, \quad 1\ ) \\
x_{25} & = & (\ 0Mj, \quad 1, \quad 1, \quad 0Ls\ ) \\
x_{26} & = & (\ 0Mj, \quad 0Mj, \quad 1, \quad 0Ls\ ) \\
x_{27} & = & (\ 1, \quad 0Mj, \quad 0Mj, \quad 0Ls\ ) \\
x_{28} & = & (\ 0Ls, \quad 0Ls, \quad 0Mj, \quad 1\ ) \\
x_{29} & = & (\ 0Ls, \quad 0Ls, \quad 1, \quad 0Mj\ ) \\
x_{30} & = & (\ 0Ls, \quad 0Ls, \quad 0Mj, \quad 0Mj\ ) \\
x_{31} & = & (\ 1, \quad 0Ls, \quad 0Ls, \quad 0Mj\ ) \\
x_{32} & = & (\ 0Mj, \quad 0Mj, \quad 0Ls, \quad 0Ls\ )
\end{array}
$$

přičemž stav x_0 je počátečním stavem systému a stavy x_{15} až x_{32} jsou stavy koncové.

4.8 Vytvoření prostoru rozhodnutí

Pokud se podíváme do tabulek 2.1 a 2.2, tak v nich zjišťujeme, že letadla potřebují pro svůj start nebo přistání 60, 120 nebo 180 sekund. Rozhodnutí tedy budou říkat: přiděl letadlu 60, 120 nebo 180 sekund pro určitou ranvej. Protože bude známo, které zdroje náleží ke které ranveji, tak tato rozhodnutí mohou být vždy taková, že přidělí zdroje jen jedné ranveji.

Prostor rozhodnutí U bude definovat jakým způsobem přidělovat zdroje tak, aby letadlům byly přiřazeny časové intervaly 60, 120 nebo 180 sekund. Pokud by tabulky 2.1 a 2.2 vypadaly jinak (měli více prvku, byly uvedeny jiné minimální rozestupy), tak se níže popisovaný algoritmus vytváření prostoru rozhodnutí změní.

V této části budeme používat Základní časovou jednotku. Po přepočtu výše uvedených časů na tyto jednotky budou letadla potřebovat pro svůj start nebo přistání 1, 2 nebo 3 základní časové jednotky. Toto platí v případě, že používáme základní časovou jednotku o velikosti 60 sekund, což jsme definovali v podkapitole *4.3 Velikost základní časové jednotky*.

Nějaké rozhodnutí u^i tedy bude říkat přiděl konzumentovi, tedy letadlu, určitý počet zdrojů jedné ranveje.

V námi definované úloze tedy bude každé rozhodnutí přidělovat buď jeden, dva nebo tři zdroje určitému letadlu. Množina U tedy bude obsahovat všechny takové u_i, které přiřazují letadlu jeden, dva nebo tři zdroje.

Pokud je počet zdrojů P, tak se mohutnost množiny U rovna $\binom{1}{P} + \binom{2}{P} + \binom{3}{P}$, popořadě je to počet kombinací, kdy se letadlu přiřazují jeden, dva nebo tři zdroje. Mohutnost této množiny je velmi velká.

V podkapitole 4.3 je definováno jedno důležité omezení. Toto omezení říká následující: pokud budou daným spotřebitelem použity zdroje $a^i, a^j i\ < j$, tak musí být spotřebitelem použity všechny zdroje a^k takové, že platí $i\ <\ k\ <\ j$. Toto omezení velmi zmenšuje počet možných rozhodnutí. Nebudou se totiž používat všechny kombinace možných zdrojů, ale jen takové, pro které platí výše uvedené podmínka.

Ve výsledku bude mít množina U mohutnost rovnou $P + (P-1) + (P-2)$, popořadě je to počet možných rozhodnutí, která splňují výše uvedenou podmínku

a letadlu přiřazují jeden, dva nebo tři zdroje. Je zřejmé, že počet rozhodnutí je v případě s podmínkou mnohem menší.

Dále označme proměnou $(us)_i$ hodnotu t_i prvního přiděleného zdroje rozhodnutím u_i

Příklad

Mějme úlohu L obsahující tato letadla $L_1, L_2, L_3, L_4, L_5, L_6$. Přičemž L_1, L_2 jsou typu Heavy, L_3, L_4 jsou typu Medium a L_5, L_6 jsou typu Light.

Maximální potřebný počet zdrojů P jsme vypočítali v příkladě z podkapitoly 4.7.1, kde vyšlo $P = 14$.

Pro tuto úlohu bude množina U obsahovat následujících 39 rozhodnutí

Rozhodnutí přidělit jeden zdroj budou tyto

$$
\begin{array}{lcl}
u_1 & = & (\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_2 & = & (\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_3 & = & (\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_4 & = & (\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_5 & = & (\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_6 & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_7 & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_8 & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_9 & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{10} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{11} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0,\ 0\ ), \\
u_{12} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0,\ 0\ ), \\
u_{13} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 0\ ), \\
u_{14} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1\ ),
\end{array}
$$

Rozhodnutí u_1 znamená přidělit první zdroj, rozhodnutí u_2 znamená přidělit druhý zdroj, atd.

Rozhodnutí přidělit dva zdroje budou tyto

$$
\begin{array}{lcl}
u_{15} & = & (\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{16} & = & (\ 0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{17} & = & (\ 0,\ 0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{18} & = & (\ 0,\ 0,\ 0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{19} & = & (\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{20} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{21} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{22} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{23} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{24} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 0,\ 0,\ 0\ ), \\
u_{25} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 0,\ 0\ ), \\
u_{26} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 0\ ), \\
u_{27} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1\ ),
\end{array}
$$

Rozhodnutí u_{15} znamená přidělit první a druhý zdroj, rozhodnutí u_{16} znamená přidělit druhý a třetí zdroj, atd.

Rozhodnutí přidělit tři zdroje budou tyto

$$
\begin{array}{lcl}
u_{28} & = & (\ 1,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{29} & = & (\ 0,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{30} & = & (\ 0,\ 0,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{31} & = & (\ 0,\ 0,\ 0,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{32} & = & (\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{33} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{34} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{35} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{36} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0,\ 0\ ), \\
u_{37} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 1,\ 0,\ 0,\ 0\ ), \\
u_{38} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 1,\ 0,\ 0\ ), \\
u_{39} & = & (\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 1,\ 1,\ 1,\ 0\ ),
\end{array}
$$

Pro ukázku definice $(us)_i$ vybereme tato dvě rozhodnutí:
Při rozhodnutí u_{15} je $(us)_{15} = 1$, při u_{21} je $(us)_{21} = 7$

4.9 Vytvoření zobrazení T

Úloha rozdělení zdrojů obsahuje stejný počet transformačních funkcí jako je počet spotřebitelů, v našem případě letadel. Je to z důvodu, že každá transformační funkce přiděluje určitému stavu a rozhodnutí nový přípustný stav (zároveň tímto způsobem určuje, která rozhodnutí mohou být provedena). Označme tyto funkce $T^1, \ldots, T^h$, kde $L_1, \ldots, L_h$ jsou letadla v úloze. Transformační funkce bude zároveň aplikovat některá uvažovaná omezení úlohy. Například omezení na délku cesty letadla ze stojánky k ranveji.

Nechť je systém S ve stavu x^i, a je provedeno rozhodnutí u^i. Pak transformační funkce T^i převede systém do nového stavu x^{i+1} za splnění všech níže uvedených podmínek:

4.9.1 Podmínka 1

Nechť rozhodnutí u^i obsahuje rozhodnutí o použití zdrojů $a_k, \ldots, a_l$, stav x^i obsahuje na pozicích zdrojů $k, \ldots, l$ nevyužitý zdroj. Zároveň stav x^{i+1} se liší od stavu x^i přesně na těch zdrojích, které rozhodnutí u^i obsahuje a zároveň je u zdrojů ve stavu x^{i+1} uvedeno, že byly přiřazeny určité dvojici (typ letadla, směr odletu). Přičemž do skupiny této dvojice letadlo L_i náleží. Při splnění všech těchto předpokladů převede transformační funkce systém ze stavu x^i do stavu x^{i+1}

Příklad

Nechť máme úlohu L obsahující letadla L_1, L_2. Přičemž L_1 je typu Light startující na sever a L_2 je typu Medium startující na jih. V této úloze se vyskytují dvě různé dvojice (typ letadla, směr odletu). Jsou to (Light, sever) a (Medium, jih). Budeme je označovat zkratkami Ls a Mj. Maximální počet zdrojů pro tuto úlohu je $P = 4$

Vybereme nějaký stav x^i např.

$$x^i \;=\; (\;\; 1,\;\; 1,\;\; 1,\;\; 1\;\;),$$

Následně je vybráno rozhodnutí

$$u^i \;=\; (\;\; 1,\;\; 1,\;\; 0,\;\; 0\;\;),$$

Transformační funkce T^i převede systém ze stavu x^i do stavu x^{i+1}, kterým je stav

$$x^{i+1} \;=\; (\;\; 0Ls,\;\; 0Ls,\;\; 1,\;\; 1\;\;),$$

který je přípustný (viz. *příklad pro množina X všech stavů systému S* z podkapitoly 4.7).

$$\begin{array}{lcl} x^i & = & (\;\; 1, \quad 1, \quad 1, \;\; 1\;\;) \\ u^i & = & (\;\; 1, \quad 1, \quad 0, \;\; 0\;\;) \\ \downarrow & T^i & \downarrow \quad \downarrow \quad \downarrow \;\; \downarrow \\ x^{i+1} & = & (\;\; 0Ls, \;\; 0Ls, \;\; 1, \;\; 1\;\;). \end{array}$$

Jako příklady transformací, které nejsou přípustné, a tedy nebudou funkcí T^i provedeny lze uvést následující

$$\begin{array}{lcl} x^i & = & (\;\; 0Mj, \;\; 1, \quad 1, \;\; 1\;\;) \\ u^i & = & (\;\; 1, \quad 1, \quad 0, \;\; 0\;\;) \\ \downarrow & T^i & \;\; ? \quad \downarrow \quad \downarrow \;\; \downarrow \\ x^{i+1} & = & (\;\; ?, \quad 0Ls, \;\; 1, \;\; 1\;\;) \end{array}$$

$$\begin{array}{lcl} x_i & = & (\;\; 1, \quad 0Mj, \;\; 1, \;\; 1\;\;) \\ u^i & = & (\;\; 0, \quad 0, \quad 0, \;\; 1\;\;) \\ \downarrow & T^i & \downarrow \quad \downarrow \quad \downarrow \;\; \downarrow \\ x^{i+1} & = & (\;\; 0Ls, \;\; 0Ls, \;\; 1, \;\; 1\;\;) \end{array}$$

První transformace se nemůže provést, protože není ve stavu x_i dostatek volných časových jednotek na to, aby se mohla transformace provést. V druhém případě neleží výsledný stav x^{i+1} mezi přípustnými stavy X.

4.9.2 Podmínka 2

Druhá podmínka transformační funkce omezuje transformaci na stavy, které mohou být danému letadlu přiřazeny s ohledem na to, kdy dané letadlo může k ranveji přijet. Pro každé letadlo je v podkapitole 2.4 definována proměná $(tc)_i$. Transformační funkce pak převede stav x^i do stavu x^{i+1}, při rozhodnutí u^i právě tehdy když bude platit $(tc)_i < (us)_i$.

Příklad

Nechť máme úlohu nadefinovanou stejně jako v minulém příkladě této podkapitoly. Nechť pro letadlo L_i platí, že $(tc)_i = 2$

Vybereme nějaký stav x^i

$$x^i = (\ 1,\ 1,\ 1,\ 1\ ),$$

pak pokud by bylo vybráno rozhodnutí

$$u^i = (\ 1,\ 1,\ 0,\ 0\ ),$$

tak transformační funkce T^i nemůže přenést systém ze stavu x^i do stavu x^{i+1}, protože rozhodnutí nesplňuje podmínku 2.

Pokud by rozhodnutí bylo například takové

$$u^i = (\ 0,\ 0,\ 1,\ 1\ ),$$

tak se transformace T^i provést může a bude vypadat takto:

$$\begin{array}{lcl} x^i & = & (\ 1,\ 1,\ 1,\ \ 1\ \ ) \\ u^i & = & (\ 0,\ 0,\ 1,\ \ 1\ \ ) \\ \downarrow & T^i & \ \ \downarrow\ \ \downarrow\ \ \downarrow\ \ \ \downarrow \\ x^{i+1} & = & (\ 1,\ 1,\ 0Ls,\ 0Ls\ ). \end{array}$$

4.9.3 Podmínka 3

Nechť rozhodnutí u^i obsahuje rozhodnutí o použití určitého počtu časových jednotek. Označme tento počet r_i. Transformační funkce T^i převede systém ze stavu x^i do stavu x^{i+1} při rozhodnutí u^i právě tehdy, když bude platit, že $r_i < Q_i$, kde Q_i je definováno v podkapitole 4.7.1 jako maximální počet nutných časových jednotek pro letadlo L_i.

Tato podmínka zjednodušeně říká, že letadlo nemůže dostat více časových jednotek než maximálně potřebuje.

4.10 Definice cílové funkce

V optimálního rozdělení zdrojů počítáme funkci

$$F_N\left(x_1, \ldots, x_N, u_1, \ldots, u_N\right) = \sum_{i=1}^{N} G_i\left(x_i, u_i\right)$$

Přičemž se tuto funkci smažíme maximalizovat nebo minimalizovat.

Proto v každém kroku musíme znát funkci $G_i; i \in \{1, \ldots, h\}$, kde h je počet letadel v úloze, a tedy i počet transformací.

V naše případě konkrétní problém vede na úlohu vícektriteriálního programování, kde pro nalezení jejího kompromisního řešení používáme lexikografickou metodu. Dané cílové funkce seřadíme podle důležitosti do tvaru

$$(\sum_{i=1}^{N} G_i, \sum_{i=1}^{N} H_i^1, \sum_{i=1}^{N} H_i^2, \sum_{i=1}^{N} H_i^3, \sum_{i=1}^{N} H_i^4).$$

Následně je pak hledáno maximum přes všechny přípustné strategie. V některých kapitolách není nutno všechny funkce H potřeba uvažovat, a zároveň může být jejích pořadí změněno pro speciální letišti.

4.10.1 Základní funkce G_i

Cílová funkce $\sum G_i$ bude v úloze maximalizována.

Funkce G_i bude funkce počítající počet ušetřený zdrojů daným rozhodnutím oproti maximálnímu počtu zdrojů, které dané letadlo může nárokovat. Protože transformační funkce T_i nedovoluje pravidlem číslo 3 přidělení více časových jednotek, než letadlo maximálně potřebuje, tak počet ušetřených časových jednotek nemůže být záporný. Zároveň bude tato funkce aplikovat preferenci ranveje definovanou v podkapitole *2.7 Omezení provozu na dané ranveji*,

$$G_i(x^i, u^i) \quad = \quad \frac{(p_i - r_i)}{\alpha^i}$$

kde r^i je počet zdrojů přidělený daným rozhodnutím u^i, p_i je maximální počet nutných zdrojů pro letadlo L_i. Hodnota α^i se získá následujícím způsobem. V tabulce definované v podkapitole *2.7 Omezení provozu na dané ranveji* nalezneme takovou hodnotu, která odpovídá času $(us)_i$ a typu daného letadla.

4.10.2 Funkce H_i^1 pro minimalizaci odletových časů

Pro funkci $\sum -H_i^1$ budeme hledat maximum přes všechny přípustné strategie úlohy.

Funkce H_i^1 počítá časy, kdy letadlo bude odlétat. Čím dříve dostane přidělený zdroj, tím dříve odletí. Bude platit:

$$H_i^1(x^i, u^i) \quad = \quad (us)_i.$$

Příklad

Nechť máme úlohu L obsahující letadla L_1, L_2. Přičemž L_1 je typu Light startující na sever a L_2 je typu Medium startující na jih. V této úloze se vyskytují dvě různé dvojice (typ letadla, směr odletu). Jsou to (Light, sever) a (Medium, jih). Budeme je označovat zkratkami Ls a Mj. Maximální počet zdrojů pro tuto úlohu je $P = 4$.

Nechť máme tyto dvě strategie rozhodnutí:

Strategie 1

$$\begin{aligned} u^1 &= (\ 1,\ 0,\ 0,\ 0\ ), \\ u^2 &= (\ 0,\ 1,\ 0,\ 0\ ). \end{aligned}$$

Strategie 2

$$\begin{array}{rcl} u^1 & = & (\ 0,\ 1,\ 0,\ 0\ ), \\ u^2 & = & (\ 0,\ 0,\ 1,\ 0\ ). \end{array}$$

Pro obě tyto strategie je hodnota cílová funkce $\sum_{i=1}^{h} G_i\left(x_i, u_i\right)$ rovna 2, což je i maximum přes všechny přípustné strategie úlohy. Tím jsme se dostali do stavu, že máme více strategií, které dávají stejnou hodnotu cílové funkce. Přesto se musí rozhodnout, která z těchto strategií je pro provoz letiště lepší. Z logického hlediska je to první strategie, protože při její aplikaci odletí obě letadla z letiště dříve než při druhé strategii. Funkce H^1 toto logické hledisko převede do matematického pohledu. Proto bude funkce $H^1 = \sum_{i=1}^{h} H_i^1\left(x_i, u_i\right)$ v první strategii rovná 3, v druhé strategii je rovna 5. Protože budeme hledat $\max -H^1$, tak v tomto příkladě je to hodnota pro první strategii.

4.10.3 Funkce H_i^2 pro preferenci dřívějších letadel

Pro funkci $\sum H_i^2$ budeme hledat maximum přes všechny přípustné strategie.

Letadla se do úlohy přidávají v pořadí v jakém se přihlásila na TWR, že jsou ve stavu Ready. Proto také budou letadla číslována v tomto pořadí vzestupně. Je vhodné, aby letadlo, které se přihlásilo dříve, mělo možnost odletět dříve, k tomuto rozlišení je sestavena funkce H_i^2. Bude proto platit

$$H_i^2(x^i, u^i) \quad = \quad ((us)_i)^i$$

Příklad

Nechť máme úlohu L obsahující letadla L_1, L_2. Přičemž L_1 je typu Light startující na sever a L_2 je typu Medium startující na jih. V této úloze se vyskytují dvě různé dvojice (typ letadla, směr odletu). Jsou to (Light, sever) a (Medium, jih). Maximální počet zdrojů pro tuto úlohu je $P = 4$

Nechť máme tyto dvě strategie rozhodnutí

Strategie 1

$$\begin{array}{rcl} u^1 & = & (\ 1,\ 0,\ 0,\ 0\ ), \\ u^2 & = & (\ 0,\ 1,\ 0,\ 0\ ). \end{array}$$

Strategie 2

$$\begin{array}{rcl} u^1 & = & (\ 0,\ 1,\ 0,\ 0\ ), \\ u^2 & = & (\ 1,\ 0,\ 0,\ 0\ ). \end{array}$$

Pro obě tyto strategie je cílová funkce $\sum_{i=1}^{h} G_i\left(x_i, u_i\right)$ rovna hodnotě 2, což je i maximum přes všechny přípustné strategie úlohy. Zároveň bude funkce $H^1 = \sum_{i=1}^{h} H_i^1\left(x_i, u_i\right)$, dávát pro obě strategie stejný výsledek, který bude roven 3. Protože letadlo které se přihlásilo TWR dříve mělo by mít přednost, tak funkce $H^2 = \sum_{i=1}^{h} H_i^2\left(x_i, u_i\right)$ pro první strategii bude mít hodnotu 5 a pro druhou strategii 3. Protože hledáme $\max H^2$ tak bude vybrána první strategie.

4.11 Získání ideálního řešení úlohy

Ideálním řešením úlohy rozumíme takovou strategii, která končí v koncovém stavu, tedy znamená startem všech letadel za předepsaných bezpečnostních podmínek. A hodnota množiny jejích cílových funkcí je lexikograficky největší.

4.12 Kdy úloha nemá řešení

Úloha nemá řešení v případě, že alespoň jedna transformační funkce nemá definovanou transformaci. V úloze definované v této kapitole tento stav (neexistence ideálního řešení) nastat nemůže. Může nastat v rozšíření úlohy v kapitole *6 Přidání letadel se slotem*, kdy by se mohlo stát, že letadlo nemůže k ranveji přijet v takovém čase, aby stihlo odstartovat ve slotu, který mu byl přidělen. Protože se hledání ideální strategie definuje jen v této kapitole, tak by bylo vhodné hned na tento možný problém upozornit. V případě, že úloha optimálního rozdělení zdrojů nemá řešení, tak lze v některých případech použít algoritmus definovaný v kapitole *8 Předbíhání letadel na ranveji určené pro start*.

4.13 Příklad

V tomto příkladě budeme ukazovat úlohy, která bude obsahovat jen dvě letadla. Sice je taková úloha velmi malá, ale ve větších úlohách by se čtenář velmi rychle ztratil. Hlavním cílem tohoto příkladu je ukázat, jakým způsobem se úloha změní, pokud místo dvou různých skupin letadel, budou jen dvě letadla ze stejné skupiny. Mějme tedy úlohu L^1 a úlohu L^2.

Úloha L^1 obsahuje letadla L_1, L_2. Přičemž L_1 je typu Light startující na sever, L_2 je typu Medium startující na jih. Dále platí $(tc)_1 = 1, (tc)_2 = 1$. Aktuální čas je 9:57. V této úloze se vyskytují dvě různé dvojice (typ letadla, směr odletu). Jsou to (Light, sever) a (Medium, jih). Budeme proto používat zkratky Ls, Mj.

Úloha L^2 obsahuje letadla L_1, L_2. Přičemž L_1, L_2 jsou typu Light startující na sever. Dále platí $(tc)_1 = 1, (tc)_2 = 1$. Aktuální čas je 9:57. V této úloze se vyskytuje jedna dvojice (typ letadla, směr odletu). Je to (Light, sever). Budeme proto používat zkratku Ls. V této úloze by nemusela být používaná žádná zkratka, ale pro porovnání úloh je ji vhodné použít.

Pro obě úlohy bude maximální počet zdrojů
$P = p_1 + p_2 + \max_{1 \leq i \ \leq h} tc_i - \min_{1 \leq i \ \leq h} tc_i = 2 + 2 + 1 - 1 = 4$, $t = 1$

V obou úlohách se tedy budou vyskytovat zdroje a_1, a_2, a_3, a_4, Každý zdroj bude obsahovat jednu základní časovou jednotku. Každý zdroj má definovanou hodnotu t_i. Protože hodnota $t > 0$, tak je úloha posunuta v čase o t cyklů. Bude tedy platit přičemž platí t_1 = 9:58, t_2 = 9:59, t_3 = 10:00, t_4 = 10:01. V obou případech je vhodné si všimnout, proč je výhodné počítat hodnotu t. Pokud by nebyla počítána, tak by úlohy měly 5 zdrojů a zdroje by byly přiřazeny časům 9:57 - 10:01, přičemž by bylo zřejmé,že zdroj s přiřazeným časem 9:57 nemůže být využit.

Pro obě úlohy bude stejná množina rozhodnutí U. V obou příkladech se vyskytují jen letadla, která potřebují maximálně dvě základní časové jednotky. Proto také rozhodnutí budou přidělovat maximálně dvě základní časové jednotky.

Množina U bude obsahovat tyto rozhodnutí

$$\begin{array}{lcl} u_1 & = & (\ 1,\ 0,\ 0,\ 0\ ) \\ u_2 & = & (\ 1,\ 1,\ 0,\ 0\ ) \\ u_3 & = & (\ 0,\ 1,\ 0,\ 0\ ) \\ u_4 & = & (\ 0,\ 1,\ 1,\ 0\ ) \\ u_5 & = & (\ 0,\ 0,\ 1,\ 0\ ) \\ u_6 & = & (\ 0,\ 0,\ 1,\ 1\ ) \\ u_7 & = & (\ 0,\ 0,\ 0,\ 1\ ) \end{array}$$

Rozdělení průběhu úloh

Výše definovaná data pro úlohy byla pro obě úlohy stejná. Všechny následující definice jsou již pro každou úlohu různé. Proto úlohy oddělíme a budeme vytvářet každou zvlášť.

Úloha L^1

Dále se musí definovat množina možných stavů M. V definicích množiny možných stavů je základní rozdíl mezi úlohou L^1 a L^2

Pro úlohu L^1 bude množina M následující:

$$\begin{array}{lcl} x_0 & = & (\ 1,\ 1,\ 1,\ 1\ ) \\ x_1 & = & (\ 0Ls,\ 1,\ 1,\ 1\ ) \\ x_2 & = & (\ 1,\ 0Ls,\ 1,\ 1\ ) \\ x_3 & = & (\ 1,\ 1,\ 0Ls,\ 1\ ) \\ x_4 & = & (\ 1,\ 1,\ 1,\ 0Ls\ ) \\ x_5 & = & (\ 0Ls,\ 0Ls,\ 1,\ 1\ ) \\ x_6 & = & (\ 1,\ 0Ls,\ 0Ls,\ 1\ ) \\ x_7 & = & (\ 1,\ 1,\ 0Ls,\ 0Ls\ ) \\ x_8 & = & (\ 0Mj,\ 1,\ 1,\ 1\ ) \\ x_9 & = & (\ 1,\ 0Mj,\ 1,\ 1\ ) \\ x_{10} & = & (\ 1,\ 1,\ 0Mj,\ 1\ ) \\ x_{11} & = & (\ 1,\ 1,\ 1,\ 0Mj\ ) \\ x_{12} & = & (\ 0Mj,\ 0Mj,\ 1,\ 1\ ) \\ x_{13} & = & (\ 1,\ 0Mj,\ 0Mj,\ 1\ ) \\ x_{14} & = & (\ 1,\ 1,\ 0Mj,\ 0Mj\ ) \\ x_{15} & = & (\ 0Ls,\ 0Mj,\ 1,\ 1\ ) \\ x_{16} & = & (\ 0Ls,\ 1,\ 0Mj,\ 1\ ) \\ x_{17} & = & (\ 0Ls,\ 1,\ 1,\ 0Mj\ ) \\ x_{18} & = & (\ 0Ls,\ 0Mj,\ 0Mj,\ 1\ ) \\ x_{19} & = & (\ 0Ls,\ 1,\ 0Mj,\ 0Mj\ ) \\ x_{20} & = & (\ 1,\ 0Ls,\ 0Mj,\ 1\ ) \\ x_{21} & = & (\ 1,\ 0Ls,\ 1,\ 0Mj\ ) \\ x_{22} & = & (\ 1,\ 0Ls,\ 0Mj,\ 0Mj\ ) \\ x_{23} & = & (\ 1,\ 1,\ 0Ls,\ 0Mj\ ) \end{array}$$

$$
\begin{array}{lcl}
x_{24} & = & (\ 0Mj,\ 0Mj,\ 0Ls,\ 1\ ) \\
x_{25} & = & (\ 0Mj,\ 1,\ 1,\ 0Ls\ ) \\
x_{26} & = & (\ 0Mj,\ 0Mj,\ 1,\ 0Ls\ ) \\
x_{27} & = & (\ 1,\ 0Mj,\ 0Mj,\ 0Ls\ ) \\
x_{28} & = & (\ 0Ls,\ 0Ls,\ 0Mj,\ 1\ ) \\
x_{29} & = & (\ 0Ls,\ 0Ls,\ 1,\ 0Mj\ ) \\
x_{30} & = & (\ 0Ls,\ 0Ls,\ 0Mj,\ 0Mj\ ) \\
x_{31} & = & (\ 1,\ 0Ls,\ 0Ls,\ 0Mj\ ) \\
x_{32} & = & (\ 0Mj,\ 0Mj,\ 0Ls,\ 0Ls\ )
\end{array}
$$

přičemž stav x_0 je počátečním stavem systému a stavy x_{15} až x_{32} jsou stavy koncové.

Další nutnou definicí je definice funkce G_1 a G_2. Důležité je si uvědomit, že čas ve které úloha probíhá je 9:57 – 10:01, přičemž dle tabulky 2.3 se právě v 10:00 mění preference na ranveji pro letadla typu Light. Protože jsme si velmi vhodně seřadili množinu rozhodnutí, tak můžeme definovat funkce G tímto způsobem:

$$
\begin{array}{lcl}
G_1(x^i, u_i) & = & \frac{(Q_i - r_i)}{2}, i \ \leq 4 \\
G_1(x^i, u_i) & = & \frac{(Q_i - r_i)}{1}, i \ > 4 \\
G_2(x^i, u^i) & = & \frac{(Q_i - r_i)}{1}
\end{array}
$$

Poslední definici, kterou je nutno pro úlohu vytvořit, je definice transformační funkce. V podkapitole *4.9 Vytvoření zobrazení T* byla popsána pravila pro možné transformace. V tomto příkladě si je můžeme vypsat všechny transformace jako seznam. Důležité je si uvědomit, že ze stavu x_0 se lze dostat jen do stavů x_1 až x_{14} a z těchto stavů se lze dostat jen do stavů x_{15} až x_{32}

$$
\begin{array}{rcl}
x_1 & = & T_1(x_0, u_1) \\
x_2 & = & T_1(x_0, u_3) \\
x_3 & = & T_1(x_0, u_5) \\
x_4 & = & T_1(x_0, u_7) \\
x_5 & = & T_1(x_0, u_2) \\
x_6 & = & T_1(x_0, u_4) \\
x_7 & = & T_1(x_0, u_6) \\
x_{15} & = & T_1(x_9, u_1) \\
x_{16} & = & T_1(x_{10}, u_1) \\
x_{17} & = & T_1(x_{11}, u_1) \\
x_{18} & = & T_1(x_{13}, u_2) \\
x_{19} & = & T_1(x_{14}, u_1) \\
x_{20} & = & T_1(x_{10}, u_3) \\
x_{21} & = & T_1(x_{11}, u_3) \\
x_{22} & = & T_1(x_{14}, u_3) \\
x_{23} & = & T_1(x_{11}, u_5) \\
x_{24} & = & T_1(x_{12}, u_5) \\
x_{25} & = & T_1(x_8, u_7) \\
x_{26} & = & T_1(x_{12}, u_7) \\
x_{27} & = & T_1(x_{13}, u_7) \\
x_{28} & = & T_1(x_{10}, u_2)
\end{array}
$$

$$
\begin{aligned}
x_{29} &= T_1(x_{11}, u_2) \\
x_{30} &= T_1(x_{14}, u_2) \\
x_{31} &= T_1(x_{11}, u_4) \\
x_{32} &= T_1(x_{12}, u_6) \\
x_8 &= T_2(x_0, u_1) \\
x_9 &= T_2(x_0, u_3) \\
x_{10} &= T_2(x_0, u_5) \\
x_{11} &= T_2(x_0, u_7) \\
x_{12} &= T_2(x_0, u_2) \\
x_{13} &= T_2(x_0, u_4) \\
x_{14} &= T_2(x_0, u_6) \\
x_{15} &= T_2(x_1, u_3) \\
x_{16} &= T_2(x_1, u_5) \\
x_{17} &= T_2(x_1, u_7) \\
x_{18} &= T_2(x_1, u_4) \\
x_{19} &= T_2(x_1, u_6) \\
x_{20} &= T_2(x_2, u_5) \\
x_{21} &= T_2(x_2, u_7) \\
x_{22} &= T_2(x_2, u_6) \\
x_{23} &= T_2(x_3, u_7) \\
x_{24} &= T_2(x_3, u_2) \\
x_{25} &= T_2(x_4, u_1) \\
x_{26} &= T_2(x_4, u_2) \\
x_{27} &= T_2(x_4, u_4) \\
x_{28} &= T_2(x_5, u_5) \\
x_{29} &= T_2(x_5, u_7) \\
x_{30} &= T_2(x_5, u_6) \\
x_{31} &= T_2(x_6, u_7) \\
x_{32} &= T_2(x_7, u_2)
\end{aligned}
$$

Tímto je nadefinováno vše potřebné pro řešení úlohy optimálního dělení zdrojů, a úloha L^1 může být řešena za pomocí metod dynamického programování.

Úloha L^2

Pro úlohu L^2 bude množina M následující:

$$\begin{array}{lcl}
x_0 & = & (\ 1, \quad 1, \quad 1, \quad 1\ ) \\
x_1 & = & (\ 0Ls, \quad 1, \quad 1, \quad 1\ ) \\
x_2 & = & (\ 1, \quad 0Ls, \quad 1, \quad 1\ ) \\
x_3 & = & (\ 1, \quad 1, \quad 0Ls, \quad 1\ ) \\
x_4 & = & (\ 1, \quad 1, \quad 1, \quad 0Ls\ ) \\
x_5 & = & (\ 0Ls, \quad 0Ls, \quad 1, \quad 1\ ) \\
x_6 & = & (\ 1, \quad 0Ls, \quad 0Ls, \quad 1\ ) \\
x_7 & = & (\ 1, \quad 1, \quad 0Ls, \quad 0Ls\ ) \\
x_8 & = & (\ 0Ls_1, \quad 0Ls_2, \quad 1, \quad 1\ ) \\
x_9 & = & (\ 0Ls_1, \quad 1, \quad 0Ls_2, \quad 1\ ) \\
x_{10} & = & (\ 0Ls_1, \quad 1, \quad 1, \quad 0Ls_2\ ) \\
x_{11} & = & (\ 0Ls_1, \quad 0Ls_2, \quad 0Ls_2, \quad 1\ ) \\
x_{12} & = & (\ 0Ls_1, \quad 1, \quad 0Ls_2, \quad 0Ls_2\ ) \\
x_{13} & = & (\ 1, \quad 0Ls_1, \quad 0Ls_2, \quad 1\ ) \\
x_{14} & = & (\ 1, \quad 0Ls_1, \quad 1, \quad 0Ls_2\ ) \\
x_{15} & = & (\ 1, \quad 0Ls_1, \quad 0Ls_2, \quad 0Ls_2\ ) \\
x_{16} & = & (\ 1, \quad 1, \quad 0Ls_1, \quad 0Ls_2\ ) \\
x_{17} & = & (\ 1, \quad 0Ls_2, \quad 0Ls_2, \quad 0Ls_1\ ) \\
x_{18} & = & (\ 0Ls_1, \quad 0Ls_1, \quad 0Ls_2, \quad 1\ ) \\
x_{19} & = & (\ 0Ls_1, \quad 0Ls_1, \quad 1, \quad 0Ls_2\ ) \\
x_{20} & = & (\ 0Ls_1, \quad 0Ls_1, \quad 0Ls_2, \quad 0Ls_2\ )
\end{array}$$

Další nutnou definicí je definice funkce G_1 a G_2. Důležité je si uvědomit, že čas ve které úloha probíhá je 9:57 - 10:01,přičemž dle tabulky 2.3 se právě v 10:00 mění preference na ranveji pro letadla typu Light. Protože jsme si velmi vhodně seřadili množinu rozhodnutí, tak můžeme definovat funkce G tímto způsobem:

$$\begin{array}{lcl}
G_1(x^i, u_i) & = & G_2(x^i, u_i) \quad = \quad \frac{(Q_i - r_i)}{2}, i \ \leq 4 \\
G_1(x^i, u_i) & = & G_2(x^i, u_i) \quad = \quad \frac{(Q_i - r_i)}{1}, i \ > 4
\end{array}$$

Poslední definici, kterou je nutno pro úlohu vytvořit, je definice transformační funkce. V podkapitole *4.9 Vytvoření zobrazení T* byla popsána pravila pro možné transformace. V tomto příkladě si je můžeme vypsat všechny transformace jako seznam. Důležité je si uvědomit, že ze stavu x_0 se lze dostat jen do stavů x_1 až x_7 a z těchto stavů se lze dostat jen do stavů x_8 až x_{20}

$$\begin{array}{lcl}
x_1 & = & T_1(x_0, u_1) \\
x_2 & = & T_1(x_0, u_3) \\
x_3 & = & T_1(x_0, u_5) \\
x_4 & = & T_1(x_0, u_7) \\
x_5 & = & T_1(x_0, u_2) \\
x_6 & = & T_1(x_0, u_4) \\
x_7 & = & T_1(x_0, u_6) \\
x_8 & = & T_1(x_1, u_3) \\
x_8 & = & T_1(x_2, u_1)
\end{array}$$

$$
\begin{aligned}
x_9 &= T_1(x_3, u_1) \\
x_9 &= T_1(x_1, u_5) \\
x_{10} &= T_1(x_4, u_1) \\
x_{10} &= T_1(x_1, u_7) \\
x_{11} &= T_1(x_1, u_4) \\
x_{12} &= T_1(x_1, u_6) \\
x_{13} &= T_1(x_2, u_5) \\
x_{13} &= T_1(x_3, u_3) \\
x_{14} &= T_1(x_2, u_7) \\
x_{14} &= T_1(x_4, u_3) \\
x_{15} &= T_1(x_2, u_6) \\
x_{16} &= T_1(x_3, u_7) \\
x_{16} &= T_1(x_4, u_5) \\
x_{17} &= T_1(x_6, u_7) \\
x_{18} &= T_1(x_5, u_5) \\
x_{19} &= T_1(x_5, u_7) \\
x_{20} &= T_1(x_5, u_6)
\end{aligned}
$$

přičemž stav x_0 je počátečním stavem systému a stavy x_8 až x_{20} jsou stavy koncové.
Tímto je nadefinováno vše potřebné pro řešení úlohy optimálního dělení zdrojů, a úloha L^2 může být řešena za pomocí metod dynamického programování.

Kapitola 5

Řešení problému včetně pojíždějících letadel

V kapitole *4 Nejjednodušší úloha optimálního rozdělení zdrojů pro letiště* se definuje úloha, která nezahrnuje v sobě letadla která jsou na cestě mezi stojánkou a ranvejí. V této kapitole se bude popisovat rozšíření úlohy definované v kapitole 4.

Pojíždějící letadla mění svůj stav v úloze z kapitoly 4 tak, že dochází ke změně jejich času, kdy mají určený start. Tedy změnou přidělených zdrojů oproti minulé úloze.

Na obrázku 5.1 jsou z obrázku 2.3 vybrána jen ta letadla, která jsou v této kapitole použita. Zároveň bylo provedeno jejich přečíslování tak, aby odpovídalo číslování v této kapitole.

Jsou to tato letadla:

všechna letadla stojící na svých stojánkách. Na obrázku 5.1 jsou to letadla L_1, L_2

všechna letadla mezi stojánkou a řadou před ranvejí. Na obrázku 5.1 letadlo L_3

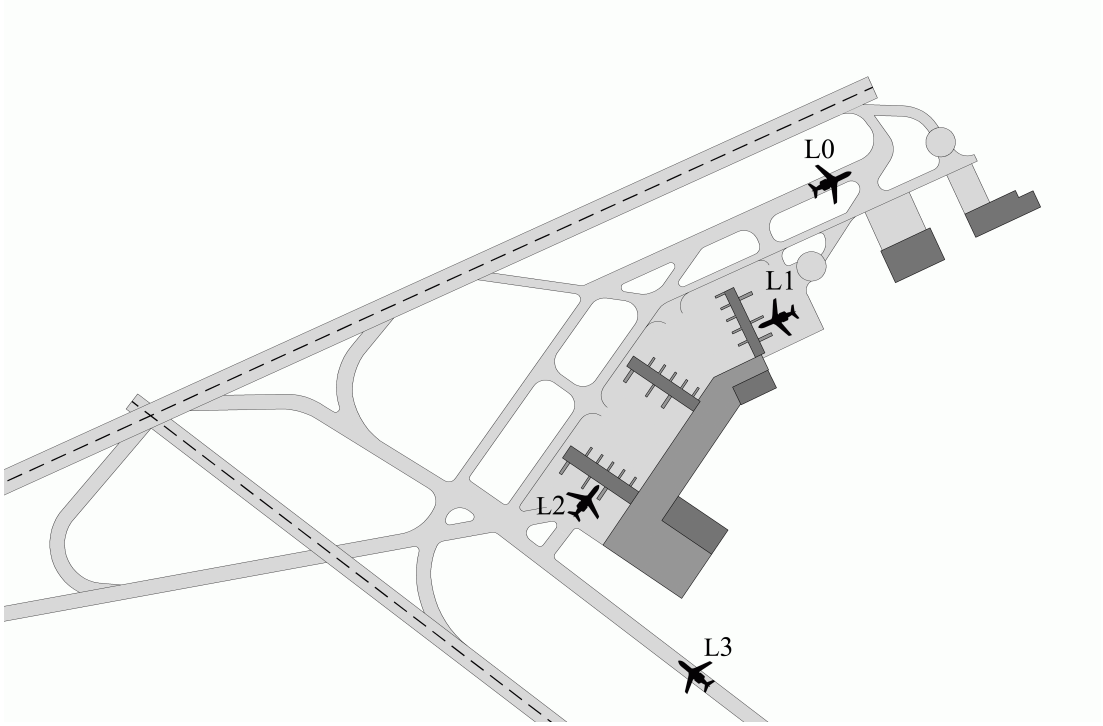
Poslední letadlo v řadě před ranveji. (případně může být ranvej volná). Na obrázku 4.1 je to letadlo L_0

Oproti úloze z kapitoly 4 se změní velmi málo. Jediná změna je v definici hodnoty $(tc)_a$.

Hodnota $(tc)_a$ bude v této kapitole označovat čas, který by letadlu trvala cesta z aktuálního místa do řady před ranvejí. Tato hodnota zůstává pro letadlo stojící na stojánce shodná s hodnotou definovanou v části 4.9. Pokud letadlo ze stojánky již vyjede, tak lze tuto hodnotu s velmi velkou přesností určovat[1].

Touto přeformulací hodnoty $tc_i, i = 1, \ldots, h$, jsme vytvořili úlohu, která je stejná jako úloha v kapitole 4. Bohužel tato nová definice $(tc)_a$ není s ohledem na praktický provoz ideální. Nová definice jen přidá pojíždějící letadla do úlohy

[1]Pro sledování umístění letadel na ploše letiště se používá pozemní radar, z jeho výstupů lze dopočítat, jak dlouho bude ještě letadlu trvat cesta. Toto je teoretická úvaha, v praxi ji ještě nikdo neimplementoval



Obrázek 5.1: Obrázek letiště obsahující letadla, která se budou v této úloze používat

dynamického programování. Cílem práce je mimo jiné minimalizovat čas, po který letadlu na zemi běží motory, aby zbytečně nespotřebovávalo palivo.

Z tohoto důvodu je nutno rozšířit definicí z podkapitoly *4.10 Definice cílové funkce*. Přidáme další funkci H.

5.0.1 Funkce H^3 pro preferenci pojíždějících letadel

Pro funkci $\sum -H_i^3$ budeme hledat maximum přes přípustné strategie úlohy.

Definujme hodnotu $(us)_i^p$ jako původní první přidělený zdroj pro letadlo, které je na cestě mezi stojánkou a ranvejí. Jinak řečeno $(us)_i^p$ určuje, kdy mělo letadlo podle předchozí řešené úlohy odletět. V této úloze to bude konstanta.

Pro ostatní letadla definujme $(us)_i^p = (us)_i$

Následně pro funkci H_i^3 bude platit následující vztah:

$$H_i^3(x^i, u^i) \quad = \quad (us)_i - (us)_i^p.$$

Pokud letadlo stojí na stojánce, případně je na cestě ze stojánky na ranvej a úloha mu dle ostatních funkcí přidělila stejné zdroje jako v minulé úloze, tak se hodnoty $(us)_i, (us)_i^p$ rovnají. Pokud by bylo letadlo již mezi stojánkou a ranvejí a byly mu zdroje oproti minulé úloze přeplánovány, tak bude mít funkce kladou hodnotu.

5.0.2 Preference letiště

V této a minulé kapitole jsme nadefinovali cílovou funkci G a několik pomocných funkcí H^i. Pro všechna letiště je primárním požadavkem, aby letadlo netrávilo na ranveji více času než je nezbytně nutné. Pro tento požadavek je definována funkce G_i, ta byla také byla proto nazvána funkcí hlavní. Úloha je následně řešena jako hledání lexikografického maxima na množině funkcí

$$(\sum_{i=1}^{N} G_i, -\sum_{i=1}^{N} H_i^1, -\sum_{i=1}^{N} H_i^3, \sum_{i=1}^{N} H_i^2)$$

přes všechny přípustné strategie.

Všechny letiště se zde neshodou na preferencich. Některá letiště preferují, aby letadlo odletěla co nejdříve, některá preferují, aby se letadla, co již nastartovala motory a pohybuji se mezi stojánkou a ranvejí zbytečně nezdržovala, protože tím jen zbytečně pálí palivo. V prvním případě by letiště používalo pořadí funkcí

$$(\sum_{i=1}^{N} G_i, -\sum_{i=1}^{N} H_i^1, -\sum_{i=1}^{N} H_i^3, \sum_{i=1}^{N} H_i^2),$$

v druhém případě by to bylo pořadí

$$(\sum_{i=1}^{N} G_i, -\sum_{i=1}^{N} H_i^3, -\sum_{i=1}^{N} H_i^1, \sum_{i=1}^{N} H_i^2).$$

V algoritmech z podkapitoly *4.11 Získání ideálního řešení úlohy* se nic nezmění kromě tohoto pořadí použití funkcí H^i.

Kapitola 6

Přidání letadel se slotem

Tato kapitola navazuje na úlohu definovanou v kapitolách *4 Nejjednodušší úloha optimálního rozdělení zdrojů pro letiště* a *5 Řešení problému včetně pojíždějících letadel*.

Jedinou změnou je přidání slotů pro různá letadla. Přidání slotu znamená omezení odletu letadla na určitou dobu. Omezení časů odletů letadel je možno provést přidáním určité podmínky do definice transformační funkce T, která je definována v podkapitole 4.9

Za pomoci projekce, definované v části *4.6 Projekce času*, můžeme definovat pro letadlo L_i čas počátku slotu jako ps_i a konce jako pk_i. Jinak řečeno definujeme první a poslední zdroj, který letadlo může využít. Důležité je uvědomit si, že letadlo, které již poslalo příkaz Ready, může mít daný slot přidělen až za určitou dobu, a ve chvíli, kdy by úloha počítala s P zdroji, definoványmi v podkapitole *4.7.1 Maximální počet zdrojů pro danou úlohu* a platilo by, že $P < ps_a$, tak by úloha nemohla najít žádnou strategii.
Existuje možnost řešení tohoto problému, při které se nemusí měnit algoritmy na vytváření úloh definované v kapitolách 4 a 5. Jde o

zvětšení délky úlohy. Porovnáme maximální délku dané úlohy P s $(ps_a + p_a)$ a větší hodnota bude použita jako maximální délka úlohy. Problémem je, že v úloze by mohl velmi narůst počet zdrojů.

Druhá možnost je, že letadlo, které nesplňuje podmínku $ps_a < P$ odstraníme z dané úlohy, a přiřadíme ho až do následně řešené úlohy. Tento způsob je nejvhodnější protože zbytečně nezvětšuje počet zdrojů. Proto ukážeme, jakým způsobem se vytváření našich úloh změní.

1. Nalezneme P pro zadaná letadla.
2. Pro letadla L_a, která mají přidělený slot[1] zkontrolujeme, zdali $ps_a \leq P$,
   (a) pokud ano. Tak jsme nalezli P takové, že každému letadlu může být v úloze přidělen zdroj.

[1] sloty jsou přidělovány jen některým letadlům. Tedy je nemusí mít všechna

(b) v opačném případě vyjmeme první letadlo nesplňující tuto nerovnost z množiny letadel pro aktuálně řešenou úlohu rozdělení zdrojů. Letadlo bude následně přidáno do úlohy, která se bude počítat později. A přejdeme na krok 1.

Tato část nepotřebuje heuristické zkoumání, protože tímto algoritmem zmenšíme aktuální úlohu rozdělení zdrojů. Zároveň toto zmenšení nebude na úkor následně řešené úlohy optimálního rozdělení zdrojů, protože v ní by vyřazená letadla stejně musela být použita.

6.0.3 Rozšíření funkce T o podmínku číslo 4

Čtvrtá podmínka bude velmi podobná podmínce číslo 2. Tato podmínka nebude platit pro všechny transformace. Bude platit jen pro takové transformace T^i, kde letadlo L_i má přidělen slot.

Čtvrtá podmínka transformační funkce omezuje transformace na takové, aby ve výsledném stavu po transformaci byly letadlu přiděleny takové zdroje, které by dané letadlo mohlo využít. Transformační funkce převede stav x^i do stavu x^{i+1}, při rozhodnutí u^i právě tehdy když bude platit $(ps)_i \leq (us)_i \leq (pk)_i$.

6.1 Neexistence řešení

Podmínky číslo 2 a 4 transformační funkce T mohou zamezovat nalezení jakékoliv strategie. Je to v případě, že letadlo nemůže stihnout dojet k ranveji dříve než mu skončí možný slot. O této možnosti jsme se zmiňovali na konci kapitoly *4 Nejjednodušší úloha optimálního rozdělení zdrojů pro letiště*. V tomto případě existují tyto možnosti. Buď požádat o nový slot, nebo v případě, že je před ranvejí řada letadel, využít možnost předbíhání, která je definovaná v kapitole *8 Předbíhání letadel na ranveji určené pro start*.

Kapitola 7

Globální pohled na úlohu dynamického programováni

V kapitole *4 Nejjednodušší úloha optimálního rozdělení zdrojů pro letiště* byla definována základní úloha rozdělení zdrojů řešící náš zjednodušený problém. V kapitolách *5 Řešení problému včetně pojíždějících letadel* a *6 Přidání letadel se slotem* bylo řešení z kapitoly 4 rozšiřováno o další podmínky. Z hlediska letištního provozu je zřejmé, že tyto úlohy budou řešeny mnohokrát ve velmi malých časových rozestupech. V této části bude rozebráno, jakým způsobem lze pro úlohu právě řešenou využít informací z již vyřešených úloh.

Vytvořme tyto předpoklady:

Každé letadlo má přiřazeno unikátní číslo. Tedy pro L_a je vždy hodnota a unikátní po celý den.

Definujeme zdroje a_i pro celý den s pevně nastaveným t_i. Každý zdroj má pevně přidělen určitý časový okamžik dne. Například a_0 je přiřazen času 0:00:00, tedy půlnoci.

Následně úloha dělení zdrojů nebude v určitý okamžik dne probíhat se zdroji $a_1, \ldots, a_P$, ale bude probíhat v intervalu $a_i, \ldots a_{i+P}$, kdy zdroj a_i odpovídá aktuálnímu okamžiku ve dne. Toto posunutí indexů nemá na řešení úlohy žádný vliv. Proto může být použito.

Protože každé letadlo má svůj jednoznačný identifikátor, tak i transformační funkce bude danému letadlu jednoznačně přiřazena. Změna očíslování žádným způsobem nezmění transformace prováděné těmito funkcemi. Proto můžeme toto přečíslování provést. Jen úloha posune indexy transformační funkce a místo transformace $T^1, \ldots, T^h$, bude používat transformace, které budou označeny $T^{i+1}, \ldots, T^{i+h}$

Maximální počet potřebných zdrojů pro úlohu zůstane stejný, protože závisí na letadlech v úloze, a ty se jen přečíslovaly.

Množina rozhodnutí U zůstává stejná, protože závisí jen na maximálním počtu potřebných zdrojů a letadlech, které do úlohy vstupují.

Definice funkce G v kapitolách 4, 5 a 6 se přečíslování letadel a zdrojů také nedotkne. Proto v nich nemusí být nic změněno.

Nová úloha je tedy ekvivalentní s úlohou definovanou v kapitole 6.

7.1 Přidání a odebrání letadla

7.1.1 Přidání letadla

Nechť máme vyřešenou úlohu, označenou L^1. Data z řešení jsou uložena. Pokud se bude řešit úloha L^2, která bude obsahovat oproti úloze L^1 navíc letadlo L_k.

Toto rozšíření úlohy L^1 na úlohu L^2 rozšiřuje množinu zdrojů o určitý počet zdrojů. Tento počet bude záviset na parametrech letadla L_k, kterými jsou p_k (což je maximální počet potřebných zdrojů pro dané letadlo) a $(tc)_i$ (což je počet základních jednotek, které trvá letadlu trvá cesta od stojánky k ranveji). Následně se musí rozšírit počet stavů systému. Pokud již v úloze L^1 bylo letadlo se stejnou dvojicí (typ letadla, směr odletu), tak nebude zvětšení počtu zdrojů tak velké,jako v případě,že tam letadlo s takovou dvojicí nebylo. Množina U se také rozšíří. Její rozšíření bude s ohledem na počet rozhodnutí v úloze L^1 malé. Důležité je, že všechny stavy, rozhodnutí a hodnoty cílových funkcí, která byly definovány v úloze L^1 lze využít i v úloze L^2. Jedinými zásadně změněné budou transformace. Většina transformací zůstane stejná, ale musí být zkontrolována podmínka číslo 2 transformační funkce. Takto budou odstraněny některé původně možné strategie, při kterých v původní úloze L^1 byla daná transformace povolena, protože mohl být letadlu přiřazen určitý zdroj, ale v úloze L^2 by letadlo tento zdroj nestihlo využít, protože by nestihlo přijet k ranveji. To je z důvodu, že úloha L^2 je počítána s určitým časovým odstupem od úlohy L^1.

Jediné hodnoty, které se musejí spočítat znovu, jsou hodnoty, pro které jsou ovlivněny novým letadlem L_k. Jednoduchost takového přepočtu by se dala jednoduše popsat následujícím způsobem (tento příklad není vůbec identický s úlohou optimálního přidělení zdrojů , ale je nejjednodušší na pochopení, jakým způsobem se zvětšuje obtížnost úlohy přidáním jednoho prvku). Nechť máme tabulku $a \times b$ a hodnoty všech políček již známe. Následně přidejme jeden sloupec a jeden řádek a pro tento přidaný řádek a sloupec spočítejme hodnoty. Je zřejmé, že přepočítaných hodnot je $a \cdot b$ a nově počítaných hodnot je $a + b + 1$.

7.1.2 Odebrání letadla

Pro dané letadlo se ze stavů systému S odstraní všechny stavy systému, které nebudou splňovat pravidla pro stav při použití nové množiny letadel (všechna letadla v úloze bez tohoto odebíraného letadla). Odstraní se nadbytečné zdroje, z množiny rozhodnutí U se odstraní všechna nadbytečná rozhodnutí, a následně se odstraní všechny hodnoty cílových funkcí, které se týkaly daného letadla. Tímto způsobem se odstraní část dat v úloze, takže zbytek úlohy se bude dát vypočítat rychleji.

7.1.3 Výhody odebíraní letadel

Přestože je úloha kdykoliv schopna dopočítávat hodnoty cílových funkcí nově přidaného letadla, je vhodné minimalizovat počet takových dopočítání. Pokud tedy nějaké letadlo již v úloze není potřeba využívat, tak je zbytečné, aby s tímto letadlem bylo počítáno. A existovaly by mezi stavy systému, takové stavy, které by se tímto letadlem zaobíraly. Proto je vhodné všechna letadla, která již v úloze nevystupují, odebírat z úlohy, protože pak se velikost a tedy i náročnost úlohy zmenšuje.

7.2 Průběžné řešení úlohy

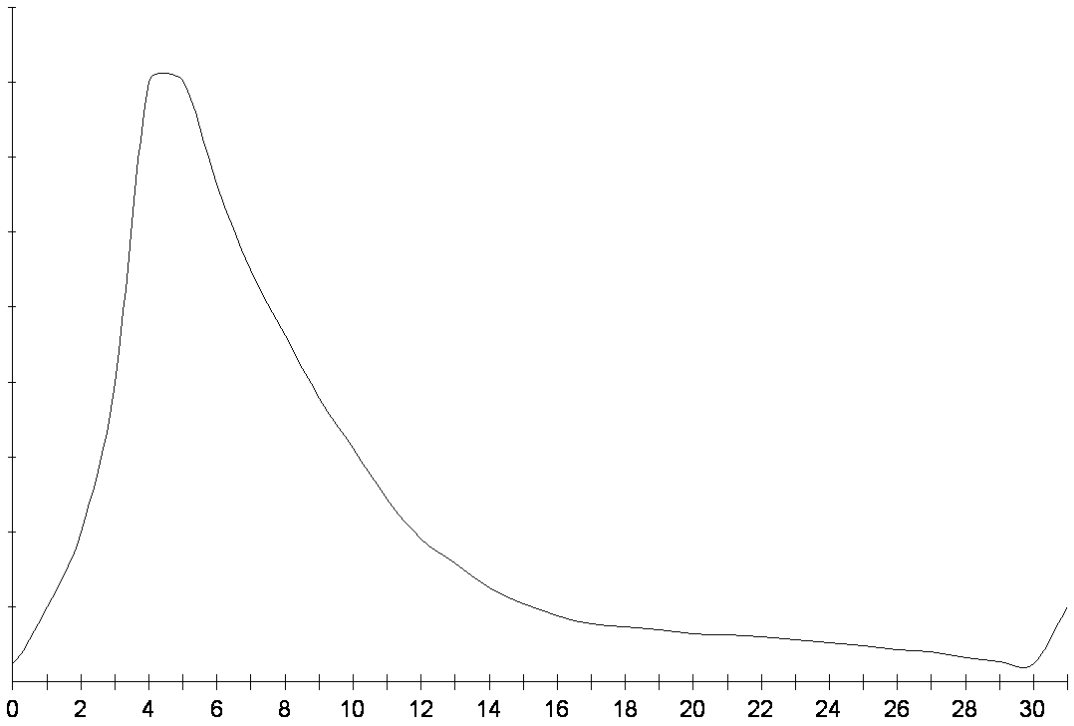
Díky možnosti operativně přidávat a odebírat letadla z množin letadel se naskytuje možnost také úlohu plynule řešit. Nemusíme totiž vždy čekat, až budou všechny hodnoty cílových funkcí pro danou množinu letadel vypočítány, ale pokud jsou již vypočítány hodnoty pro nějakou podmnožinu letadel, tak lze na těchto datech dále hledat ideální řešení. Zároveň se tímto výpočet velmi zrychluje, protože se nemusí čekat, až se vytvoří všechny tabulky.

7.3 Hlavní výhoda průběžného řešení úlohy

Zásadní výhodou tohoto přístupu je možnost předpočítání dat pro naši úlohu. Odlety letadel jsou plánovány vždy v určitém časovém intervalu, který se nazývá odletový slot. V této kapitole je používán druhý význam slovního spojení odletový slot, který je definován v *2.8 Odletový slot*. V době tvorby této práce byl tento slot pro pražské letiště dlouhý 30 minut[1]. Do daného slotu je určen maximální počet odlétajících letadel. V návrzích se počítalo s tím, že letadla která mají přidělený daný slot budou odlétat rovnoměrně po celou dobu tohoto slotu. Na obrázku 7.1 je zobrazen přibližný graf jak to v reálném světě vypadá. Většina letadel posílá informaci Ready v prvních 15 minutách, a pak následuje období klidu, kdy těchto informací Ready chodí velmi málo. Protože je známo, která letadla mají v následujícím slotu odlétat, tak si lze pro tato plánovaná letadla v období klidu předpřipravit data pro úlohu, a ve chvíli, kdy letadlo pošle oznámení ready, tak se tyto předpočítaná data mohou do úlohy přidat společně s přidávaným letadlem, a nemusejí se začít nově počítat. Dopočtení změn hodnot funkcí H je v porovnání s počítáním úlohy zanedbatelné, proto se v krátkém okamžiku po přijetí oznámení ready může v datech úlohy začít hledat optimální přídělení zdrojů a provádět výběr mezi strategiemi. Takto lze v krátkém okamžiku přidat do úlohy mnoho letadel.

Není ale vhodné takto předpočítávat hodnoty pro dlouhý časový interval a to z důvodu, že by bylo nutno počítat data pro zbytečně mnoho letadel. Jako ideální hodnota času, kdy se má začínat předpočet očekávané úlohy, pro 30-ti minutový slot, se jeví hodnota 15 minut před začátkem daného slotu. Předpočítáme si tedy

[1]Plánuje se změna na 15 minut.



Obrázek 7.1: Počty přijatých hlášení Ready v dané minutě

před daným slotem data pro úlohu s letadly, u kterých předpokládáme, že stav ready zašlou v daném slotu (například zpožděná letadla předpočítávána být nemusejí). Díky tomu bude úloha již na daná letadla připravena, protože bude mít předpočítaná data a výsledek bude moci být dodán obsluze mnohem rychleji než kdyby se úloha začala počitat až po přijetí požadavků. Většinou letadla z vypočítané úlohy také do 20-ti minut odletí z letiště. Tak budou odstraněna z úlohy a následně lze předpočítávat data pro další slot.

Kapitola 8

Předbíhání letadel na ranveji určené pro start

Od této kapitoly se nebudou používat všechny definice definované v kapitolách 3 až 7. Všechny definice budou nadefinovány znova.

V praxi existují případy, kdy není možné řešením úlohy na optimální rozdělení zdrojů nalézt žádné ideální řešení. Ve většině případů tento stav nastane ve chvíli, kdy je v úloze letadlo, které má přidělený slot, ve kterém nemůže odletět bez toho, aby nepředběhlo letadla stojící v řadě před ranvejí. Tímto vznikne nutnost přednostního zařazení tohoto mimořádného letadla do řady letadel stojících před ranvejí (viz definice v části *2.11 Dělení letadel dle různých typů úloh*). V tomto případě je nutné některá letadla čekající v řadě předběhnout a zařadit odlet tohoto mimořádného letadla mezi některá dvě letadla čekající v řadě před ranvejí při zachování požadovaných bezpečnostních odstupů mezi letadly. Pro tento účel sestavujeme několik úloh lineárního programování, pomocí kterých zjistíme, zda takové zařazení mimořádného letadla mezi některá dvě čekající letadla je možné. Účelová funkce každé této úlohy bude zkonstruována tak, že její hodnota vyjadřuje vhodnost každého přípustného řešení s ohledem na některá další kritéria, jako např. počet čekajících letadel, která byla mimořádným letem předběhnuta. Nežádoucí je vyslání dvou letadel stejným směrem a pod. Pro všechny úlohy, které budou vytvořeny na stejné skupině letadel, bude účelová funkce stejná. Proto je můžeme porovnávat, a hledat z hlediska provozu ideální zařazení mimořádného letadela. Bezpečnostní kritéria jsou již přitom zohledněna v omezení úloh.

Příklad

Mějme tuto frontu letadel $L_1, L_2, L_3, L_4, L_5, L_6$ zobrazenou na obrázku 8.1, přičemž je právě čas T

L_1 se rozjíždí na start (má startovat v čase T).

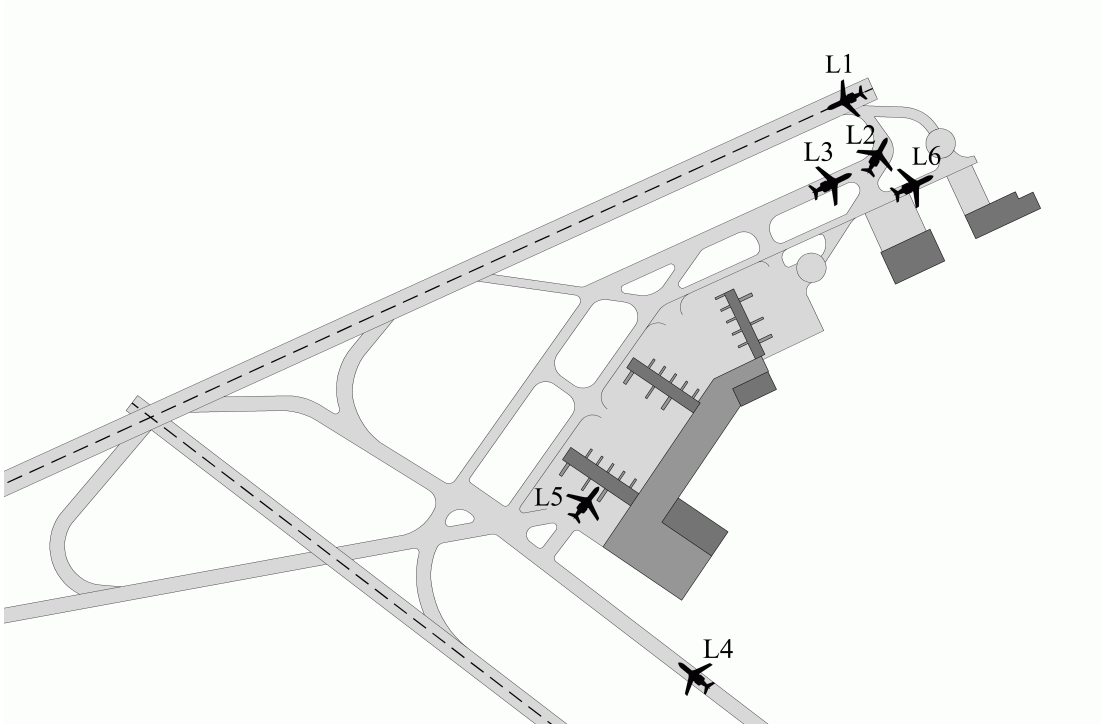
L_6 předjíždí ostatní letadla. Na ranvej přijede za 20 sekund a má startovat v čase T + 60 sekund.

L_2 čeká první v řadě. Má posunut start na čas T + 120 sekund z původních T + 60 sekund.

L_3 čeká druhé v řadě. Má posunut start na čas T + 180 sekund z původních T + 120 sekund.

L_4 pojíždí ze stojánky a do řady mu to bude trvat ještě 150 sekund. Má posunut start na čas T + 240 sekund z původních T + 180 sekund.

L_5 čeká na své stojánce, odkud mu cesta do řady trvá 160 sekund. Má posunut start na čas T + 300 sekund z původních T + 240 sekund.



Obrázek 8.1: Situační obrázek příklad pro předbíhání letadel

8.1 Definice úlohy linenárního programování

Úloha lineárního programování hledá minimum nebo maximum účelové funkce na určité množině. Všechna letadla která se budou vyskytovat v úloze lineárního programování budou mít již naplánován start. Účelová funkce bude vyjadřovat zpoždění letadel oproti tomuto plánovanému času startu, při zařazování mimořádného letadla. Množina na které bude optimální hodnota hledána bude dimenze $h+1$ .Množina bude kladným podprostorem R^{h+1}. Každá dimenze tohoto prostoru bude přiřazena jednomu letadlu z řešené úlohy. Přičemž hledaná hodnota v každé dimenzi bude značit čas startu nebo přistání daného letadla. budeme hledat $h+1$–tici, která bude splňovat určitá omezení. V případě, že množina přípustných řešení $M; M \subset R^{h+1}$ vytvořená úlohou bude neprázdná, tak vždy existuje řešení úlohy, tedy možnost zařazení mimořádného letadla, které splňuje bezpečnostní

kritéria (tj. řešení je prvkem množiny M) a je ze všech prvků množiny M nejvýhodnější (tj. dává minimální hodnotu účelové funkce úlohy). V případě, že pro všechny vytovřené úlohy platí, že $M = \emptyset$ tak žádné řešení úlohy neexistuje, a je třeba požádat o nový odletový slot pro toto mimořádné letadlo.

8.2 Matematická formulace úloh lineárního programování

Nechť máme úlohu o N letadlech. Pak můžeme následovně definovat

Množinu M

$$M = \{x \in R^N | x \geq 0, x_i + t_{i,j} \leq x_j, i \leq j, i, j \in 1, \ldots N\},$$

kde $t_{i,j}$ je minimální rozestup mezi letadly L_i a L_j v daném pořadí.

Účelovou funkci

$$min_M \sum_{i=1}^{N} x_i - (ox)_i$$

kde značí původně plánovaný čas odletu letadla x_i

Následně při dodání dalších omezení množiny M můžeme řešit úlohy lineárního programování, které budou matematicky odpovídat reálnému provozu na letišti.

8.3 Množina M přípustných řešení

Pro danou frontu a mimořádné letadlo bude každá úloha linearního programování řešena v prostoru R^{h+1}, kde $h+1$ je počet letadel v dané úloze (tedy h letadel ve frontě + jedno mimořádné letadlo). Označme letadla ve frontě $L_1, \ldots, L_h$ a mimořádné letadlo jako L_{h+1}. Pak pro každé letadlo L_a z $L_1, \ldots, L_{h+1}$ hodnota x_a bude značit čas, kdy by mělo startovat. Tento čas bude uváděn v sekundách od tzv. počátku úlohy.

Počátek úlohy definujeme jako čas, posledního použití ranveje před vytvářením této úlohy (přistávající letadlo přistálo a opustilo ranvej, případně startující letadlo odstartovalo a již ranvej neovlivňuje). K tomuto času budeme přepočítávat veškeré časové hodnoty důležité pro tuto úlohu. Důvodem takového přepočtu je to, že se bude pracovat s menšími čísly. U tohoto přepočtu lze například použít upravenou definici používanou v operačních systémech UNIX[7].

V úloze se bude hledat vektor $x = (x_1, \ldots, x_{h+1}) \in M$.
V této části se nemusíme zabývat časem, který trvá letadlům z jejich aktuální pozice do řady před ranvejí, protože zařazením letadla před ně se tento čas může jen zvýšit. Všechna již naplánovaná letadla musí být v úloze zastoupena, protože mimořádné letadlo by mohlo kterémukoliv z nich posunout čas odletu na takový čas, že by dané letadlo nemuselo stihnout svůj slot. Jediné letadlo, u kterého musí být počítáno s jeho cestou k ranveji je mimořádné letadlo.

Definice množiny M

Pro množinu M budou platit následující omezení. Pro každá dvě po sobě následující letadla L_i a L_j z $L_1, \ldots, L_{h+1}$ musí platit následující nerovnost:

$$x_i + t_{i,j} \leq x_j \quad i \in [1, \ldots, h+1], j \in [1, \ldots, h+1], i \neq j,$$
$$L_j \text{ je letadlo startující po } L_i$$

kde hodnota $t_{i,j}$ označuje v sekundách minimální časový bezpečnostní rozestup mezi letadly L_i a L_j. Tento rozestup může být vypočten z minimálního úplavu mezi těmito dvěma letadly, případně je to doba v sekundách nutná mezi starty dvou letadel do stejného směru, případně jiná bezpečnostní omezení. Vždy se používá nejvyšší hodnota z výše popsaných, která pro daná dvě letadla platí.

Pro všechna letadla L_a, která mají přidělen slot, definujeme čas počátku slotu jako $(rs)_a$ a konce slotu jako $(rk)_a$. (Nepoužívá se zde přepočet na cykly definovaný v kapitole *6 Přidání letadel se slotem*, ale čas v sekundách od počátku úlohy.) Do omezení množiny M přibudou pro každé letadlo L_a s přiděleným slotem tyto dvě nerovnosti:

$$(rs)_a \leq x_a, L_a\text{je letadlo s přiděleným slotem,}$$
$$x_a \leq (rk)_a, L_a\text{je letadlo s přiděleným slotem.}$$

Protože letadlu L_{h+1} trvá cesta od stojánky k ranveji určitý čas, tak musí být také tento čas brán v úvahu. Označme $(tc)_{h+1}$ jako čas sekundách od počátku úlohy, kdy může letadlo L_{h+1} nejdříve dojet na ranvej. Pak bude platit, že

$$(tc)_{h+1} \leq x_{h+1}$$

8.4 Účelová funkce

V této práci bude účelová funkce úlohy lineárního programování velmi jednoduchá. Lze však vytvořit i složitější funkci, která bude lépe vystihovat potřeby navigování letadel. Případně lze změnit úlohu lienárního programování na úlohu konvexního programování, což by mělo za následek změnu algoritmů výpočtu. Ale pro základní ukázku použití algoritmu na řešení našeho problému je níže definovaná účelová funkce nejvhodnější. Hodnota účelové funkce bude v tomto případě závislá jen na velikostí zpoždění ostatních letadel. Označme účelovou funkci $F(x)$ a úkolem je spočítat $\min_{x \in M} F(x)$.

Pro definici funkce $F(x)$ musíme nadefinovat původní plánovaný čas startu. Pro každé letadlo $L_a \in \{L_1, \ldots, L_h\}$ definujeme $(ox)_a$ jako, původně plánovaný čas startu v sekundách počátku času.

Definice účelové funkce

$$\begin{aligned} F(x) &= F(x_1, \ldots, x_h), \\ F(x) &:= \sum_{i=1}^{h} x_i - (ox)_i \end{aligned}$$

8.5 Výběr zařazování letadla

Na začátku této kapitoly bylo řečeno, že bude počítáno několik úlohy lineárního programování. V každé úloze bude mimořádné letadlo L_{h+1} zařazeno mezi dvě letadla z $L_1, \ldots, L_h$, po sobě využívající ranvej. Celkem je možností, kam zařadit mimořádné letadlo $h-1$. Výběr zařazování letadla má za úkol najít minimum optimálních hodnot všech těchto řešených úloh. Při námi definované účelové funkci je nejvhodnější vybírat mezi algoritmy používanými pro seřazování čísel, kterými budeme hledat frontu s nejmenším počet předběhnutých letadel, pro kterou bude platit, že všechna letadla mohou odletět. V teorii seřazování čísel je také vysvětleno, proč je pro malý počet čísel vhodné dělat přímé porovnání a pro větší je vhodnější použití metody půlení intervalů. V naší úloze jsou důvody pro výběr algoritmů stejné, proto budou představeny oba dva algoritmy a způsob, jakým by byly použity pro hledání optimálního řešení.

8.5.1 Děleni intervalu

Z ohledem na možnost paralelizace problému při počítání na strojích s více procesory (každý procesor počítá jednu úlohu lineárního programování), není pojem půlení intervalu nejvhodnější. Protože zařazování mimořádného letadla bude probíhat za pomocí dělení intervalu, tak je vhodné vysvětlit, jakým způsobem bude daný problém řešen při použití jednoho procesoru (což je problém půlení intervalu). Nechť máme h letadel ve frontě a letadlo L_{h+1} značí mimořádné letadlo. Vybereme letadla L_i, L_{i+1} tak, že $i = \left\lceil \frac{h}{2} \right\rceil$ [1]. Následně úlohu řešíme pro následující frontu letadel.

$$L_1, \ldots, L_i, L_{h+1}, L_{i+1}, \ldots, L_h$$

Pokud neexistuje řešení úlohy lineárního programování při výše uvedeném zařazení letadla L_{h+1}, tak řešme úlohu s frontou letadel $L_1, \ldots, L_h$, kde se přidá letadlo L_{h+1} mezi letadla L_j, L_{j+1} $j = \left\lceil \frac{i}{2} \right\rceil$. Po přejmenování j na i lze upravenou úlohu řešit rekurzivně. Pokud j je rovno 0, tak nelze mimořádné letadlo zařadit. V praxi to znamená, že jakékoliv zařazení mimořádného letadla do fronty letadel vytvoří problém v zařazení jiného letadla ve frontě.

Pokud existuje řešení úlohy s touto frontou letadel, tak algoritmus přejde na úlohu, která tento problém řeší pro frontu letadel $L_{i+1}, \ldots, L_h$ a přidává se letadlo L_{h+1}. Důvod, proč lze takto úlohu zmenšit je velmi jednoduchý. Při vybrané účelové funkci bude mimořádné letadlo ovlivňovat jen letadla za ním. Zároveň letadla zařazená před tímto letadlem, nemají na mimořádné letadlo žádny vliv, a ani mimořádné letadlo nemá na tyto letadla žádný vliv (letadla odletí podle svého původního plánu), proto nemusejí být v úloze uvažována.

Pokud se $i + 1 = h$, tak algoritmus bude uvažovat jen letadlo L_{h+1} a to tím způsobem, že letadlo odlétá v nejkratší možné době po letadle L_h. Pokud i tato úloha nebude mít řešení, letadlo nemůže odstartovat tak, aby splnilo všechny časové požadavky pro svůj odlet.

Algoritmus by mohl končit jedním z následujících případů:

[1] $\lceil x \rceil$ značí horní celou část čísla x

1. $j = 0$ (tedy letadlo nelze nikam zařadit);

2. $F(x) = 0$ (tedy letadlo lze zařadit za všechna letadla v řadě). Tato možnost nemůže nastat, protože by byla již vyřešena úlohou dynamického programování, kterou by letadlo L_j bylo zařazeno za letadla stojící v řadě před ranvejí;

3. V ostatních případech nalezneme takovéto dvě fronty

$$\begin{aligned} F1 &: L_1, \ldots, L_i, L_{h+1}, L_{i+1}, L_{i+2} \ldots, L_h, \\ F2 &: L_1, \ldots, L_i, L_{i+1}, L_{h+1}, L_{i+2} \ldots, L_h, \\ i &\in \{1, \ldots, h-2\} \end{aligned}$$

   Přičemž pro frontu F1 optimálního řešení existuje, a pro frontu F2 optimálního řešení neexistuje. V tomto případě je optimální zařazení mimořádného letadla L_{h+1} za letadlo L_i.

8.5.2 Paralelizace

Interval $[1, \ldots, h]$ nemusí být dělen jen na dvě části. V případě využití více procesorů lze tento interval dělit na více částí. Po vyřešení všech těchto paralelních úloh lineárního programování budeme dále hledat řešení v takovém intervalu, pro který bude platit, že pro jeho dolní hranici bylo řešení nalezeno a pro horní hranici je hodnota účelové funkce rovná 0 nebo optimální řešení neexistuje.

8.5.3 Zařazování od konce

Zařazování letadel od konce je vhodné pro méně jak 8 letadel. Při zařazování od konce je vhodné si uvědomit, jakým způsobem je definována účelová funkce v této kapitole. Není totiž nutno vždy řešit úlohu lineárního programování pro všechna letadla ve frontě, ale jen pro všechna letadla ve frontě, která jsou za mimořádně zařazeným letadlem, doplněna o poslední letadlo před mimořádně zařazeným letadlem.

Z tohoto ohledu je zařazování od konce výhodné, protože jsou nejdříve řešeny úlohy o malé dimenzi množiny M a s posunováním mimořádného letadla ve frontě směrem k jejímu počátku se dimenze množiny M zvětšuje. Zároveň při dané účelové funkci tvoří optimální řešení úloh, při posunování mimořádného letadla směrem ke konci fronty, nerostoucí posloupnost. Zároveň, pokud je nalezeno řešení úlohy tak je zřejmé, že lepší řešení úlohy existovat nebude. Proto se nemusí další případy (ty náročnější) počítat.

Pro případ možnosti využití více procesorů lze počítat více úloh lineárního programování, ve kterých je mimořádné letadlo zařazováno od konce.

8.6 Příklad pro předbíhání letadel na ranveji určené pro start

Mějme letadla L_1, L_2, L_3, L_4 které stojí v řadě před ranvejí. Letadlo L_5 nemůže být zařazeno do fronty pomocí úlohy na optimální rozdělení zdrojů, protože by nestihlo svůj slot. Proto budou řešeny úlohy lineárního programování.
Pro jednotlivá letadla budou platit tyto předpoklady.

L_1 je typu Heavy, má startovat v čase 11:10.

L_2 je typu Light, má startovat v čase 11:13.

L_3 je typu Medium, má startovat v čase 11:14. Má přidělen slot který končí v 11:15.

L_4 je typu Medium, má startovat v čase 11:15.

L_5 je typu Light, má přidělen slot který končí v 11:15, a letadlo může nejdříve přijet k ranveji v 11:12.

Možných zařazení mimořádného letadla L_5 je celkem 5. Úloha bude tedy řešit 5 úloh lineárního programování a bude hledat minimum hodnot účelových funkcí přes všechny tyto úlohy. Jako počátek úlohy označíme čas 11:10. K tomuto času bude vztahovat hodnoty ve všech úlohách, přičemž v tomto příkladě budeme používat jako jednotky minuty, místo sekund ve kterých je provedena definice.

Účelová funkce bude nadefinována pro všechny úlohy lineárního programování stejná.

$$F(x) \quad := \quad (x_1 - 0) + (x_2 - 3) + (x_3 - 4) + (x_4 - 5)$$

Budeme tedy hledat $\min_M(F(x))$, kde množiny M jsou pro úlohy definovány následovně:

Pro všechny úlohy bude platit, že $M \in R^5$ s těmito omezeními

$$\begin{aligned} x_1, x_2, x_3, x_4, x_4 &\geq 0, \\ x_5 &\geq 1, \\ x_3 &\leq 5, \\ x_5 &\leq 5, \\ x_1 + 3 &\leq x_2, \\ x_2 + 1 &\leq x_3, \\ x_3 + 1 &\leq x_4. \end{aligned}$$

Přičemž první omezení omezuje časy startů letadel jen na budoucnost, protože letadla nemohou plánovat do minulosti, druhé omezení se vztahuje k nejkratšímu možnému příjezdu letadla L_5 k ranveji, třetí a čtvrté omezení se vztahují ke slotům letadel L_3 a L_5 (počátek slotu nevyužíváme, protože nastal před začátkem této úlohy) a zbylá omezení se vztahují k povinným rozestupům mezi letadly.
Následně pro každou ze čtyř úloh budou platit ještě další omezení množiny M:

Pro úlohu s pořadím L_5, L_1, L_2, L_3, L_4 bude navíc omezení

$$x_5 + 1 \leq x_1.$$

Pro úlohu s pořadím L_1, L_5, L_2, L_3, L_4 budou navíc omezení

$$\begin{aligned} x_1 + 3 &\leq x_5, \\ x_5 + 1 &\leq x_2. \end{aligned}$$

Pro úlohu s pořadím L_1, L_2, L_5, L_3, L_4 budou navíc omezení

$$\begin{aligned} x_2 + 1 &\leq x_5, \\ x_5 + 1 &\leq x_3. \end{aligned}$$

Pro úlohu s pořadím L_1, L_2, L_3, L_5, L_4 budou navíc omezení

$$\begin{aligned} x_3 + 2 &\leq x_5, \\ x_5 + 1 &\leq x_4. \end{aligned}$$

Pro úlohu s pořadím L_1, L_2, L_3, L_4, L_5 bude navíc omezení

$$x_4 + 2 \leq x_5.$$

Díky přidaným omezením jsou některá základní omezení množiny M automaticky splněna,ale pro každou úlohu jsou to jiná omezení, proto je v tomto příkladě vhodnější je tam nechat.

Následně za pomocí simplexové metody vyřešíme těchto 5 úloh lineárního programování. Následně mezi úlohami, pro které je množina M neprázdná, vybereme tu, pro kterou vyšla v určitém bodě množiny M nejmenší hodnota účelové funkce. Následně bude pro každé letadlo L_i hodnota x_i z tohoto bodu znamenat kolik minut po počátku úlohy, tedy po 11:10, mají odstartovat. Pokud by pro všechny úlohy byla množina M prázdná, tak to znamená, že nelze zařadit mimořádné letadlo tak, že by nevznikl problém s odletovým časem jiného letadla. V takovém případě musíme požádat pro mimořádné letadlo o nový slot.

Kapitola 9

Předbíhání přistávajících letadel

V této úloze je ranvej používána nejen ke startům letadel, ale i k jejich přistávání. Přistávající letadlo je vhodné zařazovat do fronty k jejímu počátku (tedy aby letadlo přistálo co nejdříve), protože prodlužování jeho pobytu ve vzduchu je finančně nákladnější než čekání na zemi. Otázkou zůstává definice slova vhodnost. Uvažujme tento případ

Příklad

Mějme frontu letadel L_1, L_2, L_3 zobrazenou na obrázku 9.1, přičemž je právě čas T

L_1 se rozjíždí na start. Má startovat v čase T.

L_2 je typu Light, čeká v řadě jako první letadlo. Má startovat v čase T + 60 sekund.

L_3 je typu Heavy, čeká v řadě jako druhé letadlo. Má startovat v čase T + 120 sekund.

K letišti přilétá letadlo L_{11}, typu Heavy, které k ranveji přiletí nejdříve v čase T + 60 sekund. Pokud by letadlo L_{11} přistálo v čase T + 60 sekund, tak by fronta vypadala následovně:

L_1 se rozjíždí na start. Má startovat v čase T.

L_{11} je typu Heavy. Přistává v čase T + 60 sekund.

L_2 je typu Light, čeká v řadě jako první letadlo. Má posunutý start na čas T + 240 sekund z původního T + 60 sekund.

L_3 je typu Heavy, čeká v řadě jako druhé letadlo. Má posunutý start na čas T + 300 sekund z původního T + 120 sekund.

Důvod, proč letadlo L_2 změní svůj odlet z T + 60 sekund na T + 240 sekund, je nutnost dodržet odstup s ohledem na úplav po přistávajícím letadle.

Pokud by letadlo L_{11} přistálo v čase T + 120 sekund, tak by se fronta změnila následujícím způsobem:

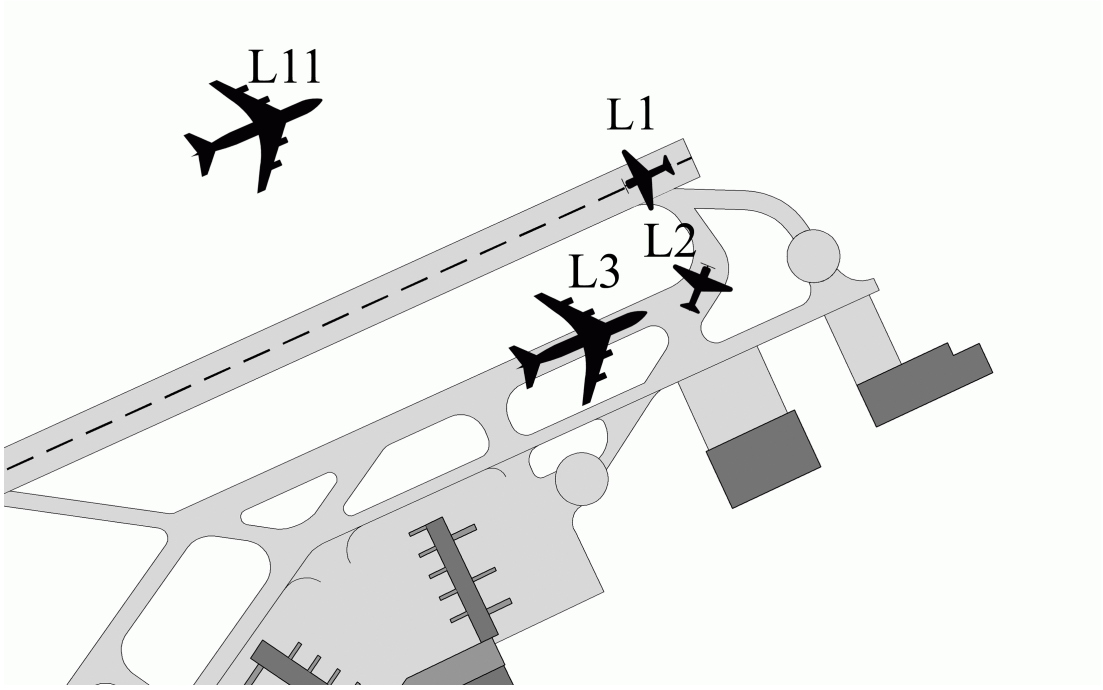
L_1 se rozjíždí na start. Má startovat v čase T.

L_2 je typu Light, čeká v řadě jako první letadlo. Má startovat v čase T + 60 sekund.

L_{11} je typu Heavy. Přistává v čase T + 120 sekund.

L_3 je typu Heavy, čeká v řadě jako druhé letadlo. Má posunutý start na čas T + 180 sekund z původního T + 120 sekund.

Pohledem na tyto dva příklady lze usoudit, že pozdržení přistání letadla L_{11} o 60 sekund dokáže velmi změnit časy odletu a přistání všech letadel z fronty. Nejlépe je tato změna vidět na odletu posledního letadla L_3. Stále zůstává otázka vhodnosti takového zdržení přistávajícího letadla.



Obrázek 9.1: Situační obrázek příklad pro předbíhání přistávajícíh letadel

---

Úloha předbíhání přistávajících letadel bude stejně jako *8 Předbíhání letadel na ranveji určené pro start* vytvářet více úloh lineárního programování, mezi nimiž budeme hledat nejmenší z optimálních hodnot účelových funkcí.

9.1 Vhodnost řešení

Vhodnost řešení této úlohy lineárního programování bude značit proměná α, která určuje vhodnost (určenou řídícím letového provozu) poměru času ušetřeného přistávajícím letadlem vůči času, o který byla ostatní letadla zpožděna. Přesnou hodnotu α lze vyzískat až z vyhodnocení výsledků z reálného provozu.

9.2 Množina M

Pro danou frontu letadel a mimořádné letadlo bude každá úloha linearního programování řešena v prostoru R^{h+1}, kde $h+1$ je počet letadel v dané úloze (tedy h letadel ve frontě + jedno mimořádné letadlo). Označme letadla ve frontě $L_1, \ldots, L_h$ a mimořádné letadlo jako L_{h+1}. Pak pro každé letadlo $L_a \in \{L_1, \ldots, L_{h+1}\}$ hodnota x_a bude značit čas, kdy by mělo letadlo L_a startovat, případně přistávat. Tento čas bude uváděn v sekundách od počátku úlohy.

V tomto případě se nemusíme zabývat časem, po který trvá cesta letadlům z jejich aktuální pozice do řady před ranvejí, protože zařazením přistávajícího letadla před ně se tento čas může jen zvýšit. Proto vždy k ranveji dojedou s předstihem. Všechna již naplánovaná letadla musí být v úloze brána v potaz, protože mimořádné letadlo by mohlo kterémukoliv z nich posunout čas odletu na takový čas, že by dané letadlo nemuselo stihnout svůj slot.

Definice množiny M

Pro každá dvě po sobě následující letadla L_i, L_j z $L_1, \ldots, L_{h+1}$ musí platit následující nerovnost:

$$x_i + t_{i,j} \leq x_j \quad i \in [1, \ldots, h+1], j \in [1, \ldots, h+1], i \neq j,$$
$$L_j \text{ je letadlo ve frontě následující po letadlu } L_i$$

kde hodnota $t_{i,j}$ označuje v sekundách minimální časový bezpečnostní rozestup mezi letadly L_i a L_j. Tento rozestup může být vypočten z minimálního úplavu mezi těmito dvěma letadly, případně je to čas nutný mezi starty dvou letadel do stejného směru, případně jiná bezpečnostní omezení. Z těchto hodnot se vždy použije ta nejvyšší. Zde je vhodné si uvědomit, že pokud je letadlo L_i startující, tak se používají hodnoty z tabulky *2.2 Minimální vzdálenosti letadel na ranveji z pohledu turbulence v úplavu pro odlétající letadla*, pokud je letadlo L_i přistávající, tak se používají hodnoty z tabulky *2.1 Minimální vzdálenosti letadel na ranveji z pohledu turbulence v úplavu pro přistávající letadla*.

Pro všechna letadla L_a, která mají přidělen slot definujeme čas počátku slotu jako $(rs)_a$ a konce jako $(rk)_a$. Nepoužívá se zde přepočet na cykly definovaný v kapitole *6 Přidání letadel se slotem*, ale čas v sekundách od počátku úlohy (počátek úlohy je definován v části 8.3). Do omezení množiny M přibudou pro každé letadlo L_a s přiděleným slotem tyto dvě nerovnosti

$$(rs)_a \leq x_a, L_a \text{je letadlo s přiděleným slotem,}$$
$$x_a \leq (rk)_a, L_a \text{je letadlo s přiděleným slotem.}$$

Protože každé přistávající letadlo při naplánování přistání začne provádět přiblížení, které již nelze přerušit, tak musí být v úloze čas jeho přistání zadán pevně (nelze ho již tedy změnit). Proto definujeme $(tc)_a$, kde $\forall L_a \in \{L_1, \ldots, L_h\}$ kde L_a je přistávající letadlo, jako počet sekund od počátku úlohy, kdy má letadlo L_a naplánováno přistání. Pak bude platit

$$(tc)_a = x_a, L_a \text{ je přistávající letadlo.}$$

Protože o čase přistání letadla L_{h+1} se rozhoduje s předstihem, tak jako hodnotu $(tc)_{h+1}$ označme počet sekund od počátku úlohy, kdy nejdříve může letadlo přistát. Následně bude platit

$$(tc)_{h+1} \leq x_{h+1}.$$

9.3 Účelová funkce

V této úloze nadefinujeme účelovou funkci, která bude záviset na čase přistání přidávaného přistávajícího letadla a čase, o který tímto přistáním přistávajícího letadla, budou letadla ve frontě posunuta.

Pro každé letadlo L_a z $L_1, \ldots, L_h$ definuje hodnotu $(px)_a$ jako čas, kdy mělo letadlo naplánován start. Pro letadlo L_{h+1} bude hodnota $(px)_{h+1}$ značit počet sekund od počátku času, kdy by letadlo mohlo nejdříve přistát na ranveji. Dále je α je koeficientem definovaným v 9.1. A β je koeficientem omezujícím vliv přistávajícího letadla na ostatní letadla.

Dále definujeme i takové, že pro frontu letadel bude platit následující

$$L_1, \ldots, L_i, L_{h+1}, L_{i+1}, \ldots, L_h$$

Definice účelové funkce

Pokud je letadlo L_{i+1} startující, pak funkce $F(x)$ je definována takto:

$$\begin{aligned} F(x) &= F(x_1, \ldots, x_{h+1}) \\ F(x) &:= (x_{h+1} - px_{h+1}) + \alpha \cdot (x_{i+1} - px_{i+1}) \cdot \sqrt[\beta]{h - (i+1)} \end{aligned}$$

Pokud je letadlo L_{i+1} přistávající, tak označme L_k takové letadlo, které je startující a je prvním ve frontě po letadlu L_{i+1}. Následně bude platit pro funkci $F(x)$ tato definice

$$\begin{aligned} F(x) &= F(x_1, \ldots, x_{h+1}) \\ F(x) &:= (x_{h+1} - px_{h+1}) + \alpha \cdot (x_k - px_k) \cdot \sqrt[\beta]{h - k} \end{aligned}$$

Hodnotu β, lze vytvořit jen na základě praktických zkušeností. Má zabránit tomu, aby při dlouhé frontě bylo mimořádně přistávající letadlo umísťováno ke konci fronty, protože α je číselná konstanta. Díky hodnotě β nebude hodnota výrazu $\alpha \cdot (x_k - px_k)$ růst lineárně s počtem přibývajících letadel, ale přírustek účelové funkce se bude zmenšovat. Hodnoty α, β mohou být pro počáteční nastavení vybrány takto

$$\begin{aligned} \alpha &= 0,5, \\ \beta &= 2. \end{aligned}$$

9.4 Výběr zařazování letadla

Na začátku této kapitoly bylo řečeno, že bude počítáno několik úlohy lineárního programování. V každé úloze bude mimořádné letadlo L_{h+1} zařazeno mezi dvě letadla z $L_1, \ldots, L_h$, která po sobě využívají ranvej. Celkem je možností kam zařadit mimořádné letadlo $h - 1$. Výběr zařazování letadla má za úkol najít minimum z optimálních hodnot všech těchto úloh. Rozhodnutí, jakým způsobem zařazovat přistávající letadlo, bude fungovat na heuristickém principu. Protože je z hlediska ekonomiky provozu letadel vhodné, aby přistávající letadlo přistálo co nejdříve, bude určitě platit $\alpha < 1$. Což v případě výše definované účelové funkce říká,že optimální hodnotu účelové funkce je vhodné hledat v případech, že přistávající letadlo bude zařazováno do fronty na místo co nejblíže počátku fronty. Proto je vhodné pro zjišťování řešení naší úlohy použít pozměněný algoritmus, popisovaný v části *8.5.3 Zařazování od konce*. V tomto případě nebude přistávající letadlo zařazováno od konce množiny letadel. Bude zařazováno od začátku množiny letadel, protože se s ohledem na definici α a účelové funkce bude minimum účelových funkcí, přes možná zařazení mimořádného letadla, nalezeno při zařazení přistávajícího letadla blíže k počátku.

Kapitola 10

Obecné předbíhání letadel

V kapitolách *8 Předbíhání letadel na ranveji určené pro start*, *9 Předbíhání přistávajících letadel* se vytváří algoritmus pro předbíhání mimořádně startujících nebo přistávajících letadel. Bohužel v kapitole 8 se počítá jen s tím, že letadla budou startovat. V kapitole 9 není zase započteno předbíhání mimořádných startujících letadel.

Dále v obou algoritmech vzniká následující problém. Předbíháním jsme do fronty zařadili mimořádné letadlo tak, aby pro dané zařazení byla hodnota příslušné účelové funkce nejmenší ze všech možných zařazení letadla. Pokud chvíli po té přijde žádost o další zařazení mimořádného letadla, které by bylo nutné zařadit před již zařazené letadlo, tak by pro danou frontu letadel přestala mít úloha řešení. Výše uvedený problém nejlépe ukazuje následující příklad.

Příklad

Mějme tuto frontu letadel L_1, L_2, L_3, L_4, L_5 zobrazenou na obrázku 10.1, kterou vytvořil algoritmus navržený v kapitole *8 Předbíhání letadel na ranveji určené pro start*, přičemž je právě čas T.

L_1 se rozjíždí na start. Má startovat v čase T.

L_2 je první v řadě. Má startovat v čase T + 60 sekund.

L_3 je druhé v řadě. Má startovat v čase T + 120 sekund.

L_4 předjíždí letadla v řadě. Má startovat v čase T + 180 sekund. Musí odstartovat do T + 200 sekund.

L_5 je třetí v řadě. Má startovat v T + 240 sekund.

K letišti přilétá letadlo L_{11}. K ranveji přiletí nejdřívev čase T + 60 sekund. Algoritmus navržený v kapitole *9 Předbíhání přistávajících letadel* by zařadil letadlo takto:

L_1 se rozjíždí na start. Má startovat v čase T.

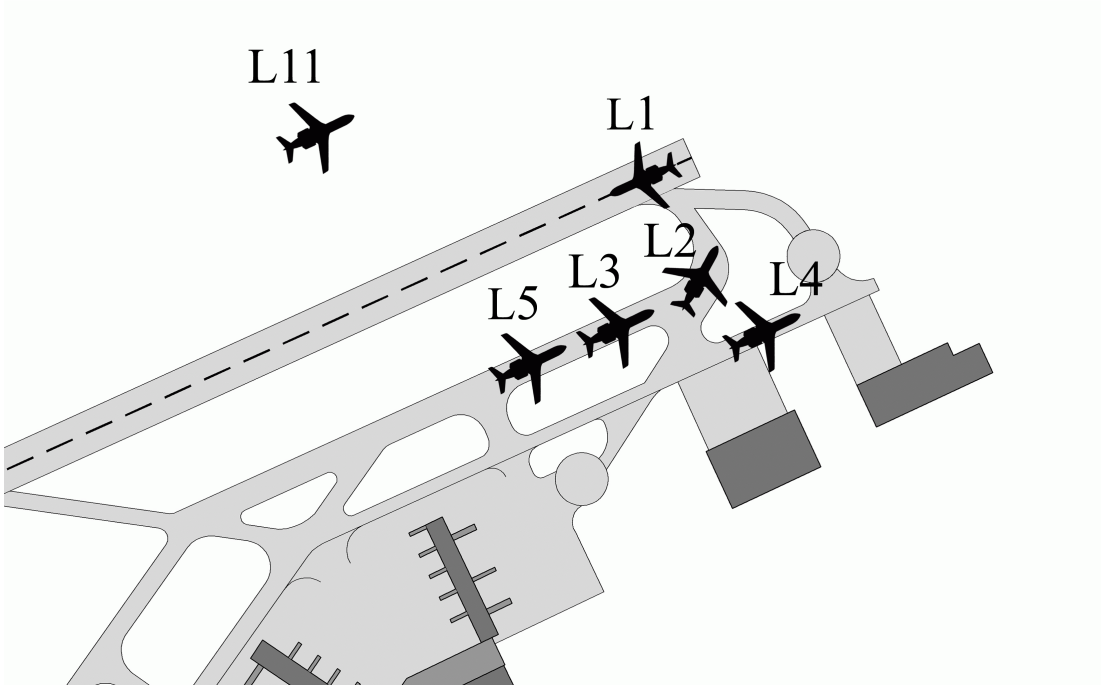
L_2 je první v řadě. Má startovat v čase T + 60 sekund.

L_3 je druhé v řadě. Má startovat v čase T + 120 sekund.

L_4 předjíždí letadla v řadě. Má startovat v čase T + 180 sekund. Musí odstartovat do T + 200 sekund.

L_{11} přistává v čase v T + 240 sekund.

L_5 je třetí v řadě. Má posunut start na čas T + 300 sekund z původního času T + 240 sekund



Obrázek 10.1: Situační obrázek

Výše uvedený případ ukazuje, co se může stát problémem v obou algoritmech definovaný v kapitolách *8 Předbíhání letadel na ranveji určené pro start* a *9 Předbíhání přistávajících letadel*. Je jím nemožnost uvažovat do algoritmu letadel, u kterých je předpoklad že se brzo zařadí do úlohy. Řešení je bohužel algoritmicky velmi složité. Muselo by se předpovědět, kdy dané letadlo přistane (což se dá s ohledem na navigaci letadel dosti přesně zjisťovat) nebo, kdy dané letadlo bude posílat zprávu Ready (což nelze žádným způsobem předpovědět, protože existuje mnoho vnějších faktorů ovlivňujících poslání této zprávy. Např. cestující přijdou pozdě).

Protože lze provést velmi přesný odhad, kdy bude letadlo přistávat, tak je možné tuto informaci ve vytváření úlohy lineárního programování využít. Informace

o předpokládaném přistání letadla je neustále aktualizována dle polohy letadla ve vzduchu a znalosti plánované trajektorie jeho letu. Následně by úloha nehledala takové pořadí letadel, pro které by byla hodnota účelové funkce nejmenší, ale hledala by posloupnost, pro kterou by byla hodnota účelové funkce taková, že by existovala jedna posloupnost taková, že její hodnota účelové funkce by byla menší. Tato posloupnost letadel by následně zajišťovala, že pokud bude dané letadlo již naplánováno ve frontě, tak i při vložení jednoho nově mimořádného letadla před toto letadlo, zůstane úloha obsahující toto letadlo ve frontě stále řešitelná a bude mít optimální hodnotu účelové funkce, tedy letadlo se ve frontě posune tak, že by při vkládání v danou chvíli (poté, co budou vložena přistávající letadla) byla hodnota účelové funkce, přes všechny možné fronty letadel, nejmenší.

Úprava předešlého příkladu

Protože při počítání algoritmu navrženého v kapitole *8 Předbíhání letadel na ranveji určené pro start* je již známo, že do doby T + 180 sekund bude přistávat jedno letadlo, tak algoritmus zařazující mimořádně startující letadlo nevybere takovou frontu, která má nejmenší hodnotu účelové funkce, ale vybere takovou frontu, jejíž hodnota účelové funkce je druhá nejmenší. Mějme frontu letadel L_1, L_2, L_3, L_4, L_5, která je zobrazenou na obrázku 10.1, přičemž je právě čas T.

L_1 se rozjíždí na start. Má startovat v čase T.

L_2 je první v řadě. Má startovat v čase T + 60 sekund.

L_4 předjíždí letadla v řadě. Má startovat v čase T + 120 sekund. Musí odstartovat do T + 200 sekund.

L_3 je druhé v řadě. Má startovat v čase T + 180 sekund.

L_5 je třetí v řadě. Má startovat v T + 240 sekund.

Protože existuje předpoklad, že se v úloze objeví vhodné letadlo pro zařazení, tak posloupnost letadel v této frontě vytváří druhou nejmenší hodnotu účelové funkce po hodnotě účelové funkce pro frontu vytvořenou v prvním příkladě této kapitoly.

K letišti přilétá letadlo L_{11}. K ranveji přiletí nejdříve čase T + 60. Algoritmus navržený v kapitole *9 Předbíhání přistávajících letadel* by zařadil letadlo takto:

L_1 se startuje v čase T.

L_{11} přistává v čase v T + 60 sekund.

L_2 je první v řadě. Má posunutý start na čas T + 120 sekund z původního T + 60 sekund.

L_4 předjíždí letadla v řadě. Má posunutý start na čas T + 180 sekund z původního T + 120 sekund. Musí odstartovat do T + 200.

L_3 je druhé v řadě. Má posunutý start na čas T + 240 sekund z původního T + 180 sekund.

L_5 je třetí v řadě. Má posunutý start na čas T + 300 sekund z původního T + 240 sekund.

Kromě této úvahy existuje druhá nutná úprava algoritmu definovaného v kapitole 8. Tato druhá úprava přidává do naší úlohy i omezení pro přistávající letadla.

10.1 Množina M

Oproti definici množiny M z kapitoly 8 změníme definici množiny M následujícím způsobem(přidáme přistávající letadla).
Označme letadla ve frontě $L_1, \ldots, L_{h+1}$, kde letadlo značené L_{h+1} je mimořádně řazené letadlo, a $L_1, \ldots, L_h$ jsou letadla ve frontě. Pak pro každé letadlo L_a z $L_1, \ldots, L_h$ bude hodnota x_a značit čas, kdy by mělo startovat, případně přistávat.
Pro každá dvě po sobě následující letadla $L_i, L_j \in \{L_1, \ldots, L_{h+1}\}$ musí platit následující nerovnost

$$x_i + t_{i,j} \leq x_j$$

kde hodnota $t_{i,j}$ označuje minimální časový bezpečnostní rozestup mezi letadly L_i a L_j. Tento rozestup může být vypočten z minimálního úplavu mezi těmito dvěma letadly, případně je to čas nutný mezi starty dvou letadel do stejného směru, případně jiné bezpečnostní omezení. Z těchto hodnot se vždy použije ta nejvyšší. Zde je vhodné si uvědomit, že pokud je letadlo L_i startující, tak se používají hodnoty z tabulky *2.2 Minimální vzdálenosti letadel na ranveji z pohledu turbulence v úplavu pro odlétající letadla*, pokud je letadlo L_i přistávající,tak se používají hodnoty z tabulky *2.1 Minimální vzdálenosti letadel na ranveji z pohledu turbulence v úplavu pro přistávající letadla*.
Pro všechna letadla L_a, která mají přidělen slot, budou přidány tyto dvě nerovnosti

$$rs_a \leq x_a, L_a\text{je letadlo s přiděleným slotem,}$$
$$x_a \leq rk_a, L_a\text{je letadlo s přiděleným slotem.}$$

Každému přistávajícímu letadlu ve frontě budou přidány tyto rovnosti

$$tc_a = x_a, L_a\text{je přistávající letadlo, kromě letadla } L_{h+1}.$$

Pro letadlo L_{h+1}, které je startující, přibude nerovnost pro určení doby cesty ze

stojánky

$$tc_{h+1} \leq x_{h+1}.$$

Pro letadlo L_{h+1}, které je přistávající, přibude nerovnost pro určení doby, kdy může nejdříve přistát

$$tc_{h+1} \leq x_{h+1}.$$

V obou případech přibude stejná nerovnost. Hodnota tc_{h+1} pro oba případy totiž znamená za jak dlouho může být letadlo nejdříve u ranveje.

10.2 Účelová funkce

L_{h+1} je startující letadlo

Účelovou funkci musíme definovat samostatně pro startující letadlo L_{h+1}. A je shodná s definicí uvedenou v kapitole *8 Předbíhání letadel na ranveji určené pro start*

$$\begin{aligned} F(x) &= F\left(x_1, \ldots, x_{h+1}\right), \\ F(x) &:= \textstyle\sum_{i=1}^{h} x_i - ox_i. \end{aligned}$$

Všechny definice proměnných jsou převzaty z kapitoly 8.

L_{h+1} je přistávající letadlo

Pro přistávající letadlo L_{h+1} bude při jeho zařazení do fronty

$$L_1, \ldots, L_i, L_{h+1}, L_{i+1}, \ldots, L_h$$

účelová funkce definována shodně s kapitolou *9 Předbíhání přistávajících letadel*
Všechny definice proměnných jsou převzaty z kapitoly 9.
Pokud je letadlo L_{i+1} startující, pak funkce $F(x)$ je definována takto

$$\begin{aligned} F(x) &= F\left(x_1, \ldots, x_{h+1}\right), \\ F(x) &:= \left(x_{h+1} - px_{h+1}\right) + \alpha \cdot \left(x_{i+1} - px_{i+1}\right) \cdot \sqrt[\beta]{h - (i+1)}. \end{aligned}$$

Pokud je letadlo L_{i+1} přistávající, tak označme L_k takové letadlo, které je startující, a je prvním ve frontě po letadlu L_{i+1}. Následně bude platit pro funkci $F(x)$ tato definice

$$\begin{aligned} F(x) &= F\left(x_1, \ldots, x_{h+1}\right) \\ F(x) &= \left(x_{h+1} - px_{h+1}\right) + \alpha \cdot \left(x_k - px_k\right) \cdot \sqrt[\beta]{h - (i+1)}. \end{aligned}$$

Změna výběru fronty

V této kapitole bylo řečeno, že z pohledu provozu nemusí být nejmenší hodnota účelové funkce z účelových funkcí, vytvořených pro stejnou množinu letadel nejvhodnější. Proto je nutno při hledání fronty na tento předpoklad myslet. Algoritmy pro výběr v jakém pořadí se budou fronty vytvořené z dané množiny letadel, mohou zůstat stejné jako v kapitolách 8 a 9, jen se vybere taková fronta, jejíž hodnota účelové funkce je n–tá nejmenší (n–tá nejmenší ze všech hodnot, které dávají ostatní fronty), pokud je předpoklad, že před řazené letadlo bude zařazeno ještě n jiných letadel.

10.2.1 Problémy řešené upraveným algoritmem

Výše uvedený algoritmus vyřeší vhodným způsobem následující příklad.

Příklad

Mějme tuto frontu letadel L_1, L_2, L_3 zobrazenou na obrázku 10.2, přičemž je právě čas T.

L_1 se rozjíždí na start. Má startovat v čase T.

L_2 je typu Light. Čeká v řadě jako první letadlo. Má startovat v čase T + 60 sekund.

L_3 je typu Light. Přistává v čase T + 180 sekund, protože se očekává zařazení jednoho mimořádného letadla. Jinak by mohlo přistát v čase T + 120 sekund.

Objevuje se letadlo L_{11}, typu Light. Potřebuje se mimořádně zařadit a odstartovat do T + 70 sekund. Obecný algoritmus zařazování mimořádného letadla vytvoří následující frontu:

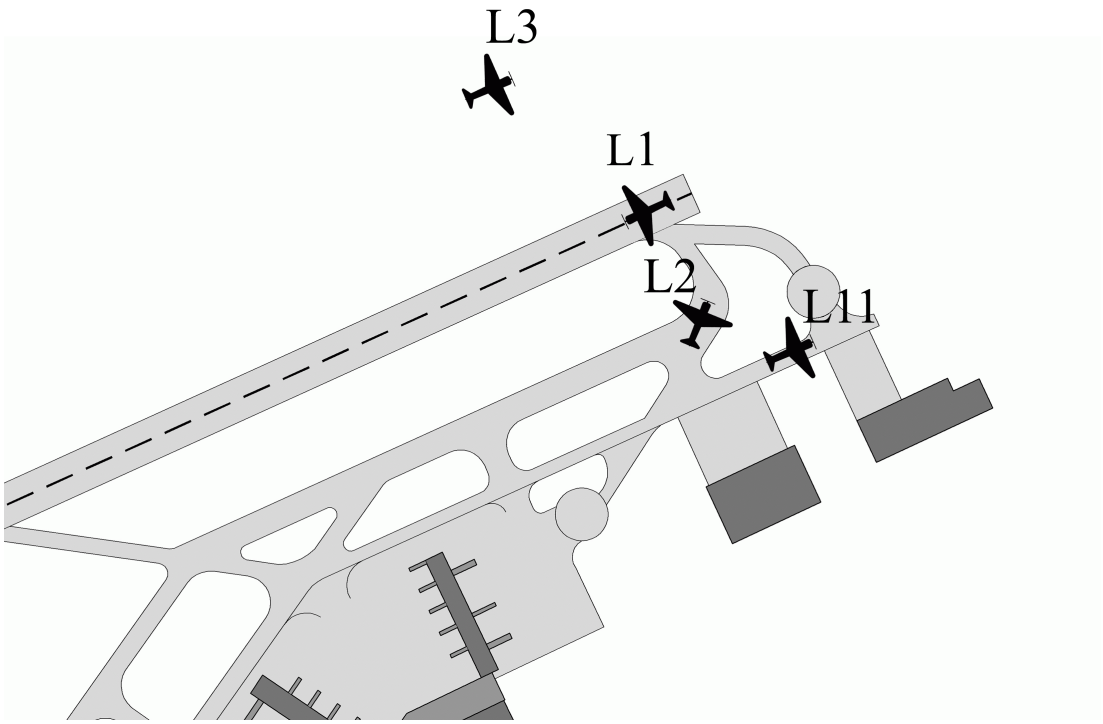
L_1 se rozjíždí na start. Má startovat v čase T.

L_{11} je typu Light. Předjíždí letadla v řadě. Má startovat v čase T + 60 sekund. Musí odstartovat do T + 70 sekund.

L_2 je typu Light. Čeká v řadě jako první letadlo. Má posunutý start na čas T + 120 sekund z původního T + 60 sekund.

L_3 je typu Light. Přistává v čase T + 180 sekund.

Bohužel následující problém, který je velmi podobný předcházejícímu, již není tímto algoritmem řešitelný



Obrázek 10.2: Situační obrázek

Příklad

Mějme tuto frontu letadel L_1, L_2, L_3 zobrazenou na obrázku 10.3, přičemž je právě čas T.

L_1 se rozjíždí na start. Má startovat v čase T.

L_2 je typu Light. Čeká v řadě jako první letadlo. Má startovat v čase T + 60 sekund.

L_3 je typu Light. Přistává v čase T + 180 sekund.

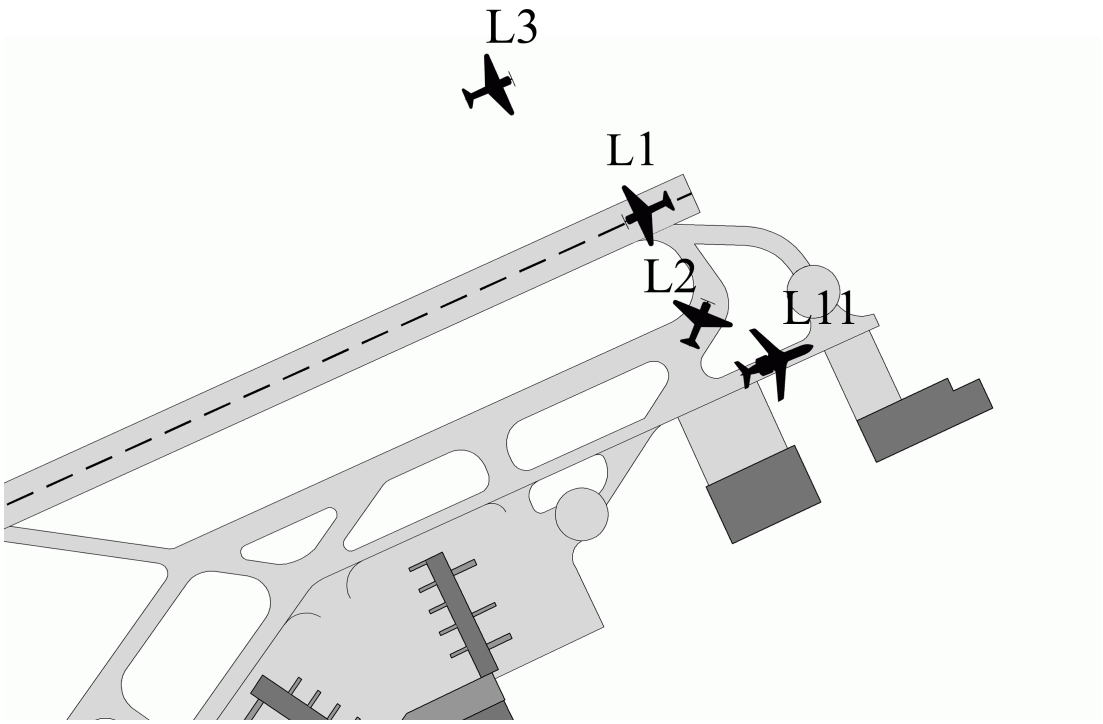
Objevuje se letadlo L_{11}, typu **Medium**. Potřebuje se mimořádně zařadit a odstartovat do času T + 70 sekund. Tento problém již algoritmus sice vyřeší, ale při složitější změně pořadí letadel ve frontě by mohlo z pohledu navigace letadel existovat řešení lepší. Vypadalo by následovně:

L_1 se rozjíždí na start. Má startovat v čase T.

L_{11} je typu Medium. Předjíždí letadla v řadě. Má startovat v čase T + 60 sekund. Musí odstartovat do T + 70 sekund.

L_3 je typu Light. Přistává v čase T + 180 sekund.

L_2 je typu Light, Čeká v řadě jako první letadlo. Má posunutý start na čas T + 240 sekund z původního času T + 60 sekund.



Obrázek 10.3: Situační obrázek

10.3 Změna řazení letadel

V části *8.5 Výběr zařazování letadla* bylo vždy posunováno ve frontě jen mimořádné letadlo. V tomto posledním uvedeném příkladě bylo provedeno menší přeuspořádání za pomocí kombinatoriky. Lze je například provést následujícím způsobem. Nechť máme tuto frontu

$$F \ : L_1, \ldots, L_i, L_{i+1}, L_{h+1}, L_{i+2}, L_{i+3}, \ldots, L_h,$$

kde všechna letadla jsou startující kromě L_{i+2}, které je přistávající, a zároveň letadlo L_{h+1} , které je mimořádně zařazované.
Kromě vyřešení úlohy pro výše uvedenou frontu F je nutno vyřešit úlohu i pro následující frontu

$$F1 : L_1, \ldots, L_i, L_{h+1}, L_{i+2}, L_{i+1}, L_{i+3}, \ldots, L_h$$

Tato změna fronty odpovídá předchozímu příkladu.
V kapitolách 8 a 9 se vždy zařazovalo jediné mimořádné letadlo do fronty, proto počet možností, jak toto letadlo zařadit byl vždy roven počtu letadel ve frontě. Tedy tato úloha měla lineární složitost. Pokud bychom při přidávání mimořádného letadla měnili pořadí jiných mimořádně přidaných letadel, tak by složitost úlohy mohla začít velmi narůstat. Pokud by doba vyřešení jedné úlohy lineárního programování trvala 1μs, tak by pro 20 letadel v úloze trvalo vyřešení všech úloh lineárního programování, které by vznikly ze všech kombinací těchto 20 letadel

zhruba 1 sekundu. Což je určitě dostatečný čas pro rozhodnutí operátora. Pro větší počet mimořádně řazených letadel by ale již muselo být používáno heuristických metod a čas výpočtu totiž narůstá exponenciálně.

10.3.1 Závěr

V této části jsme se pokusili popsat změny v algoritmech popisovaných v kapitolách 8 a 9 a vytvořit jeden algoritmus, který řeší oba problémy. Existuje ještě několik dalších situací, na které předbíhací algoritmy nedokáží správně zareagovat. Otázkou je, zdali se vyplatí vymýšlení algoritmů, které by tyto situace řešily. Hlavní otázkou v tomto rozhodování by bylo, kolik procent reálných situací by bez těchto algoritmů zůstalo nevyřešených. Proto by o jejich vytváření bylo vhodné uvažovat až po praktických zkušenostech s používáním předbíhacích algoritmů. V této kapitole jsme si několik situací popsali včetně toho, jak by mohly být řešeny. Všechno jsou to ale jen návrhy a není tedy nutno pro ně vytvářet příklady, protože příklady, které pokryjí většinu situací, které mohou v provozu nastat, byly na konci kapitol 8 a 9.

Kapitola 11

Závěr

V diplomové práci jsme se zabývali použitím metody diskrétního dynamického programování na jednu úlohu z praxe, Speciálně na úlohu optimálního rozdělení zdrojů. Bylo zde využito teorie dynamické programování, lineárního programování a vícekriteriálního programování. V kapitolách 2 až 6 je popsáno, jak vytvářet úlohy optimálního rozdělení zdrojů při podmínkách, které vyžaduje provoz letiště. V úlohách pravděpodobně nebyla aplikována všechna omezení, která mohou v reálném provozu nastat, avšak uvažované úlohy pokryjí velkou část provozem vyžadovaných omezení.

V kapitole 7 jsem navrhoval algoritmy, které by mohly problém pořadí startujících letadel na letišti řešit kontinuálně. Hlavní výhodou navrhnutého postupu je, že sice řeší postupně několik úloh optimálního rozdělení zdrojů, ale pro řešení jedné úlohy používá data vypočítaná v minulých úlohách. Tímto způsobem je zkrácen čas, který potřebují počítače pro vytváření řešení celkového provozu na letišti. Bylo také navrhnuto, jakým způsobem lze data pro úlohu předpočítávat. Tímto je dána možnost řešit úlohu i ve chvílích, kdy je na letišti nutné posunout plánovaný start letadel z důvodů kapacitních omezení letiště. Toto předpočítávání je výhodné i pro případ, kdy předpokládáme, že se v krátkém časovém intervalu přibude několik letadel, o které se náhle zvýší počet uvažovaných letadel v úloze.

Kromě metody dynamického programování existuje možnost, při které se vyzkouší všechny alternativy seřazení letadel, která při dostatečném počtu letadel by byla výhodnější než metoda dynamického programování. Zjištění toho počtu není jednoduché vzhledem k tomu je nutno brát v úvahu mnoho podmínek, které ovlivnit formulaci úlohy.

Kapitoly 8 až 10 se zabývají případy, kdy algoritmus, používající metody dynamické programování, nevygeneroval řešení. Tyto situace nastávají v případech, kdy je ranvej používaná jak pro starty, tak pro přistání letadel. Tyto případy jsou řešeny metodou lineárního programování, které lze rychle vyřešit simplexovou metodou.

Cílem této práce bylo vygenerovat algoritmy, které by byly použitelné v praxi, tedy by byly schopny rychle vytvářet řešení daných problému.

Práce vznikla na základě diskuse na konferenci s pracovníky z German Aerospace Center. A po konzultaci s pracovníky Řízení letového provozu České republiky, kteří o řešení tohoto problému měli zájem.

Literatura

[1] Nožička F.: *Dynamické programování I, Diskrétní dynamické programování*, Státní pedagogické nakladatelství, Praha, 1. vydání

[2] Grygarová L.: *Základy vícekriteriálního programování , skripta*, Univerzita Karlova, Praha 1996, 1. vydání

[3] *ICAO Annexy v českém znění L-4444 Hlava 5*, 24.11.2005, `http://lis.rlp.cz/predpisy/predpisy/index.htm`

[4] *ICAO Annexy v českém znění L-4444 Hlava 8*, 24.11.2005, `http://lis.rlp.cz/predpisy/predpisy/index.htm`

[5] *Wake turbulence* `http://en.wikipedia.org/wiki/Wake_turbulence`

[6] *Runway/Orientation and dimensions* `http://en.wikipedia.org/wiki/Runway`

[7] *UNIXový čas* `http://en.wikipedia.org/wiki/Unix_time`

[8] *Slotová koordinace Praha* `http://www.slot-czech.cz`