PORGETTO TERZO TURNO

LABORATORIO

LINGUAGGI DI SISTEMA

ANNO 2008/2009

Quartarone Giuseppe

Il progetto in esame implementa un modulo di supporto ai servizi per il turismo, ed offre delle funzioni per la pianifiazione di viaggi intermodali basati sul trasporto pubblico.
Il sistema tramite la specifica di una richiesta contenente una data, un ora di partensa, un punto di parterza ed uno di arrivo e` in grado di offrire uno o piu` tragitti sfruttando i mezzi a disposizione.
In particolare, il modulo riceve in input da riga di comando, una lista di file di configurazione, che descrivono una serie di servizi di trasporto secondo una specifica sintassi e dopo averli caricati e` in grado di rispondere alle eventuali richieste.
Considerando che il sistema sara` destinato a servire un numero assai elevato di richieste, di fondamentale importanza e` stata la schelta delle strutture dati e degli algoritmi utilizzati. Come si puo` ben notare il calcolo del tragitto puo` essere risolto come il problema di trovare un cammino su un grafo orientato, dove i nodi sono rappresentati dalle stazazioni, partenza e arrivo, e gli archi sono l'insieme dei servizi a disposizione.
La rappersentazione del grafo e` stata effettuata tramite l'utilizzo delle liste di adiacenza che ne offrono un efficiente memorizzazione, in particolare:

1 I nodi sono stati rappresentati per mezzo di una lista in cui ogni record corrispodne ad un nodo e contiene quattro campi: il nome del nodo, una chiave intera che lo identifica in modo univoco in tutto il sistema, un vettore di puntatori di sette posizioni a servizi (archi), che ne descrivono la sua stella uscente, un puntatore al successio nodo nella lista;

1 Gi archi sono rappresentati per mezzo di una lista in cui ogni record corrisponde ad un arco e contiene undici campi: il nome del servizio, il nome della sua stazione di partenza, il nome della sua stazione di arrivo, tre codici interi, di cui uno identifica il servizio, e gli altri due si riferiscono alle stazioni di arrivo e di partenza, un costo del tragitto, un orario di partenza e di arrivo, entrambi rappresentati in minuti, e per finire due puntatori, uno all'arco successivo della stella uscente cui esso appartiene ed uno al prossimo arco nella lista.

Grazie a questa rappresentazione si possono scandire agevolmente sia i nodi che gli archi, sia l'insieme degli archi di una stessa stella uscente, in oltre questa struttura mi permette di effettuare l'aggiunta di nuovi nodi e archi in modo efficiente.
Uno dei punti deboli di questo tipo di implementazione e` la  verifica dell'esistenza di un determinato nodo, infatti, per far cio` si dovrebbe scorrere sequenzialmente tutta la lista dei nodi.

Questa scelta non e` affatto applicabile, perche` il sistema potrebbe benissimo contenere migliaia di nodi. Per ovviare a questo problema, ho affiancato alle liste di adiacenza un albero binario, che memorizza i nodi in base all'ordinamento lessicografico dei loro nomi e contiene i puntatori ai singoli record della lista, in questo modo riesco ad accedere ai singoli nodi del grafo con costi ridotti in ordine di tempo.
L'albero binario e` decisamente utile durante la fase di caricamento dei dati dai file di configurazine, infatti prima di aggiungere una stazione ne controllo l'eventuale presenza ed in caso negativo, questa viene aggiunta in tesa alla lista, mentre in caso positivo questa non viene aggiunta e sfrutto il puntatore all'interno del nodo dell'albero, accedo al record contenuto nella lista e completo la sua configurazione.
Queste sono le scelte implementative utilizzate per la rappresentazione di un grafo, mentre per la risoluzione del calcolo del cammino ho scelto di utilizzare una versione adattata di Dijkstra, che fa uso di una coda con priorita`. Quest'ultima contiene i nodi che verrano visitati dall'algoritmo, ed in particolare in testa alla coda c'e` sempre il nodo con etichetta minima, infatti la coda e` implementa come un heap di minimo.
Ho scelto di utilizare un implementazione di Dijkstra per inserire tra le varie soluzioni quelle meno costose in ordine di tempo, in oltre ho supposto che, trattandosi di viaggi non ci possono essere costi negativi, cosa che mi ha scoraggiato dall'utilizzare l'algoritmo di Belman.
Dijkstra e` un algoritmo locale che permette di trovare i cammini minimi su di un grafo ciclico a costi non negativi e restituisce un vettore di predecessori che permette di ricostruire il percorso a ritroso, dalla destinazione alla sorgente. In particolare la struttura che rappersenta un predecessore e` costitiuta da quattro campi: un puntatore al nodo predecessore, un puntatore all'arco che lega il nodo con il predecessore, un etichetta ed un puntatore al nodo stesso. Tutte le etichette vengono impostate ad un valore pari a “65535”, i predecessori vengono impostati al valore del nodo sorgente, mentre i puntatori agli archi sono inizialmente pari al valore “NULL”, fa eccezione il nodo radice che ha etichetta pari a “0” e predecessore pari a “NULL”.
La particolarita` di questo algoritmo e` che sfruttando la coda con priorita` si garantisce che ogni nodo venga esaminato esattamente una volta.
L'esecuzione si sviluppa in due fasi, una di inizializzazione e una di calcolo del cammino. Per la prima fase ho fatto uso di un vettore di predecessori di lunghezza pari al numero dei nodi del grafo, e sfruttando il codice che identifica il nodo, ho messo in relazione gli indici del vettore con i nodi. In particolare se “k” e` il codice intero di un nodo, alla pozione “k-1” nel vettore dei predecessori si

trovera` il suo predecessore, questo mi permette un rapido accesso al vettore durante il calcolo del cammino.
Come precedentemente detto, Dijkstra restituisce un cammino da una sorgente ad una destinazione, quindi per calcolare un cammino alternativo, basta rieseguire l'algoritmo eliminado dal grafo un arco che precedentemente utilizzato per raggiungere la destinazione. In particolare ad ogni esecuzione elimino iterativamete gli archi facenti parte della stella uscente del nodo sorgente, inserendoli in un altro albero binario, in questo modo mi calcolo gli altri cammini modificando solo il servizio di partenza. All'iterazione successiva l'algoritmo per ogni arco, controlla se e` stato inserito nell'albero, ed in caso positivo passa all'analisi dell'arco successivo.
Un itineraio puo` essere formato da piu` servizi, e si garantisce che tra un servizio ed un altro ci siano almeno 10 minuti di differenza, in oltre il sistema restituisce le soluzione con orario di partenza maggiore o uguale all'orario contenuto nella richiesta. Il sistema non e` in grado di gestire il caso in cui l'arrivo sia nel girono successivo a quello di partenza.
Di seguito una breve descrizione delle struttre non ancora analizzate.
La struttura che caratterizza un elemento di una lista e` formata da tre campi, la chiave dell'elemento, un puntatore a void per un eventuale carico, un puntatore all'elemento successivo. Le funzioni offerte per le liste sono l'aggiunta in testa di un elemento, la relativa ricerca e la stampa dell'intera lista.
Un nodo di un albero binario invece e` formato da quatro campi, una stringa che rappresenta la chiave del nodo, un puntatore a void per un eventuale carico, ed i puntatore ai figli, destro e sinistro. Le funzioni offerte sono l'aggiunta, la cancellazione, la ricerca di un nodo, nonche` la stampa. I nodi sono memorizzati secondo l'ordine lessicografico, grazie all'utilizzo della funzione “strcmp”.
Per quanto rigurda la coda con priorita` ho sfruttato il fatto che essendo un albero completo a sinistra, si puo` rappresentare implicitamente tramite un vettore. Un nodo e` rappresentato da due campi, la chiave e l'etichetta, mentre una coda e` formata, da un vettore di nodi e altri due campi che ne indicano la dimensione massima ed il numero di elementi contenuti.
Le funzioni offerte sono, al solito, l'inserimento, l'eliminazione dell'elemento in testa, la stampa e la funzione che si occupa dell'inizializzazione. La coda e` un vettore dinamico che viene ridimensionato a seconda dei casi, in particolare se si riempie, la sua dimensione viene raddoppiata, mentre se contiene meno di n/4 elementi viene dimezzata.
Di seguito una breve descrizione delle funzioni utilizzate per la gestione del grafo, in particolare, la funzione adibita al caricamento dei file e` “readconf” che riceve in ingresso il nome del file e un puntatore al grafo. La lettura dai file e` stata effettuata tramite l'utilizzo della funzine “sscanf” che

mi permette di impostare in modo semplice ed intuitivo un formato di lettura, il quale agevola il reperimento dei dati necessari alla costruzione del grafo. Durante questa fase si effettua un minimo di cotrollo del formato del file, anche se, si suppone che i file contengano dati corretti oltre che a rispettare la giusta sintassi, in ogni modo in caso di errore il modulo viene terminato segnalando l'eventuale errore sullo standard error. Oltre a leggere le informazioni dai file, “readconf”, effettua la creazione di una relazione tra un nodo ed un arco, tramite la fuzione “addRel”. Quest'ultima al suo interno non fa altro che creare un nodo tramite “addStation”, ed associargli un arco che verra` inserito nella rispettiva lista, nonche` nella stella uscente del nodo.
La funzione che implementa l'algoritmo di Dijkstra e` “spt”, che come spiegato in precedenza utilizza sia un vettore di predecessori, sia una coda con priorita` e sia un albero binario contenente eventuali archi da scartare nel calcolo del cammino.
Le ultime due funzioni sono “cammino” e “stampasol”, che rispettivamente si occupano delle ripetute chiamate a “spt” e della stampa della soluzione. In particolare “stampasol” stampa la soluzione in modo ricorsivo, ed e` lei che inserisce gli archi da scartare nell'albero che verra` passato ad “spt”.
Di seguito un esempio di eserucuzione.

Richista mandata input:
22/12/2009|07:30|Empoli|Roma Termini

esecuzione:

Opzione 1 | 02:27
R3102 | Empoli | 08:02 | Lastra A Signa | 08:13
R3102 | Lastra A Signa | 08:14 | Firenze Rifredi | 08:27
R3102 | Firenze Rifredi | 08:28 | Firenze S. M. Novella | 08:33
AV9425 | Firenze S. M. Novella | 08:49 | Roma Termini | 10:29

Opzione 2 | 04:40
R23377 | Empoli | 08:14 | Signa | 08:27
R23377 | Signa | 08:28 | Firenze Rifredi | 08:39
R23377 | Firenze Rifredi | 08:40 | Firenze S. M. Novella | 08:47
R2307 | Firenze S. M. Novella | 09:13 | Firenze Campo Di Marte | 09:19
R2307 | Firenze Campo Di Marte | 09:20 | Figline Valdarno | 09:36
R2307 | Figline Valdarno | 09:37 | S. Giovanni Valdarno | 09:42
R2307 | S. Giovanni Valdarno | 09:43 | Montevarchi-Terranuova | 09:48
R2307 | Montevarchi-Terranuova | 09:49 | Arezzo | 10:13
R2307 | Arezzo | 10:14 | Castiglion Fiorentino | 10:26
R2307 | Castiglion Fiorentino | 10:27 | Camucia-Cortona | 10:33
R2307 | Camucia-Cortona | 10:34 | Terontola-Cortona | 10:39
R2307 | Terontola-Cortona | 10:40 | Castiglione Del Lago | 10:47
R2307 | Castiglione Del Lago | 10:48 | Chiusi-Chianciano Terme | 10:59

R2307 | Chiusi-Chianciano Terme | 11:01 | Fabro-Ficulle | 11:10
R2307 | Fabro-Ficulle | 11:11 | Orvieto | 11:26
R2307 | Orvieto | 11:27 | Alviano | 11:40
R2307 | Alviano | 11:41 | Attigliano-Bomarzo | 11:49
R2307 | Attigliano-Bomarzo | 11:50 | Orte | 12:00
R2307 | Orte | 12:02 | Roma Tiburtina | 12:44
R2307 | Roma Tiburtina | 12:46 | Roma Termini | 12:54

Opzione 3 | 02:58
R23400 | Empoli | 07:31 | Signa | 07:44
R23400 | Signa | 07:45 | Firenze Rifredi | 07:56
R23400 | Firenze Rifredi | 07:57 | Firenze S. M. Novella | 08:03
AV9425 | Firenze S. M. Novella | 08:49 | Roma Termini | 10:29

Opzione 4 | 02:42
R11708 | Empoli | 07:47 | Montelupo-Capraia | 07:52
R11708 | Montelupo-Capraia | 07:53 | Signa | 08:02
R11708 | Signa | 08:03 | Le Piagge | 08:08
R11708 | Le Piagge | 08:09 | Firenze Rifredi | 08:16
R11708 | Firenze Rifredi | 08:17 | Firenze S. M. Novella | 08:23
AV9425 | Firenze S. M. Novella | 08:49 | Roma Termini | 10:29

seconda richiesta:
21/09/2009|06:00|Pisa Centrale|Firenze S. M. Novella

Opzione 1 | 01:07
R23377 | Pisa Centrale | 07:40 | Pontedera-Casciana Terme | 07:53
R23377 | Pontedera-Casciana Terme | 07:54 | S. Miniato-Fucecchio | 08:06
R23377 | S. Miniato-Fucecchio | 08:07 | Empoli | 08:13
R23377 | Empoli | 08:14 | Signa | 08:27
R23377 | Signa | 08:28 | Firenze Rifredi | 08:39
R23377 | Firenze Rifredi | 08:40 | Firenze S. M. Novella | 08:47

Opzione 2 | 01:55
R3100 | Pisa Centrale | 06:38 | Pontedera-Casciana Terme | 06:52
R3100 | Pontedera-Casciana Terme | 06:53 | Empoli | 07:08
R3100 | Empoli | 07:09 | Lastra A Signa | 07:19
R3100 | Lastra A Signa | 07:20 | Firenze Rifredi | 07:31
R3102 | Firenze Rifredi | 08:28 | Firenze S. M. Novella | 08:33

Opzione 3 | 01:04
R3102 | Pisa Centrale | 07:29 | Pontedera-Casciana Terme | 07:44
R3102 | Pontedera-Casciana Terme | 07:45 | Empoli | 08:01
R3102 | Empoli | 08:02 | Lastra A Signa | 08:13
R3102 | Lastra A Signa | 08:14 | Firenze Rifredi | 08:27
R3102 | Firenze Rifredi | 08:28 | Firenze S. M. Novella | 08:33

Opzione 4 | 01:53
R23400 | Pisa Centrale | 06:54 | Pontedera-Casciana Terme | 07:09
R23400 | Pontedera-Casciana Terme | 07:10 | S. Miniato-Fucecchio | 07:21

R23400 | S. Miniato-Fucecchio | 07:22 | Empoli | 07:30
R23377 | Empoli | 08:14 | Signa | 08:27
R23377 | Signa | 08:28 | Firenze Rifredi | 08:39
R23377 | Firenze Rifredi | 08:40 | Firenze S. M. Novella | 08:47

Opzione 5 | 01:32
R11708 | Pisa Centrale | 07:01 | Navacchio | 07:07
R11708 | Navacchio | 07:08 | S. Frediano A Settimo | 07:10
R11708 | S. Frediano A Settimo | 07:11 | Cascina | 07:16
R11708 | Cascina | 07:17 | Pontedera-Casciana Terme | 07:24
R11708 | Pontedera-Casciana Terme | 07:25 | S. Romano-Montopoli-S. Croce | 07:33
R11708 | S. Romano-Montopoli-S. Croce | 07:34 | S. Miniato-Fucecchio | 07:38
R11708 | S. Miniato-Fucecchio | 07:39 | Empoli | 07:46
R11708 | Empoli | 07:47 | Montelupo-Capraia | 07:52
R11708 | Montelupo-Capraia | 07:53 | Signa | 08:02
R11708 | Signa | 08:03 | Le Piagge | 08:08
R11708 | Le Piagge | 08:09 | Firenze Rifredi | 08:16
R3102 | Firenze Rifredi | 08:28 | Firenze S. M. Novella | 08:33

Opzione 6 | 02:32
R23376 | Pisa Centrale | 06:01 | Navacchio | 06:07
R23376 | Navacchio | 06:08 | S. Frediano A Settimo | 06:10
R23376 | S. Frediano A Settimo | 06:11 | Cascina | 06:16
R23376 | Cascina | 06:17 | Pontedera-Casciana Terme | 06:24
R23376 | Pontedera-Casciana Terme | 06:25 | S. Romano-Montopoli-S. Croce | 06:33
R23376 | S. Romano-Montopoli-S. Croce | 06:34 | S. Miniato-Fucecchio | 06:38
R23376 | S. Miniato-Fucecchio | 06:39 | Empoli | 06:46
R23376 | Empoli | 06:47 | Montelupo-Capraia | 06:52
R23376 | Montelupo-Capraia | 06:53 | Signa | 07:02
R23376 | Signa | 07:03 | Le Piagge | 07:08
R23376 | Le Piagge | 07:09 | Firenze Rifredi | 07:16
R3102 | Firenze Rifredi | 08:28 | Firenze S. M. Novella | 08:33°