HARVESTING RELATIONAL TABLES FROM LISTS ON THE WEB

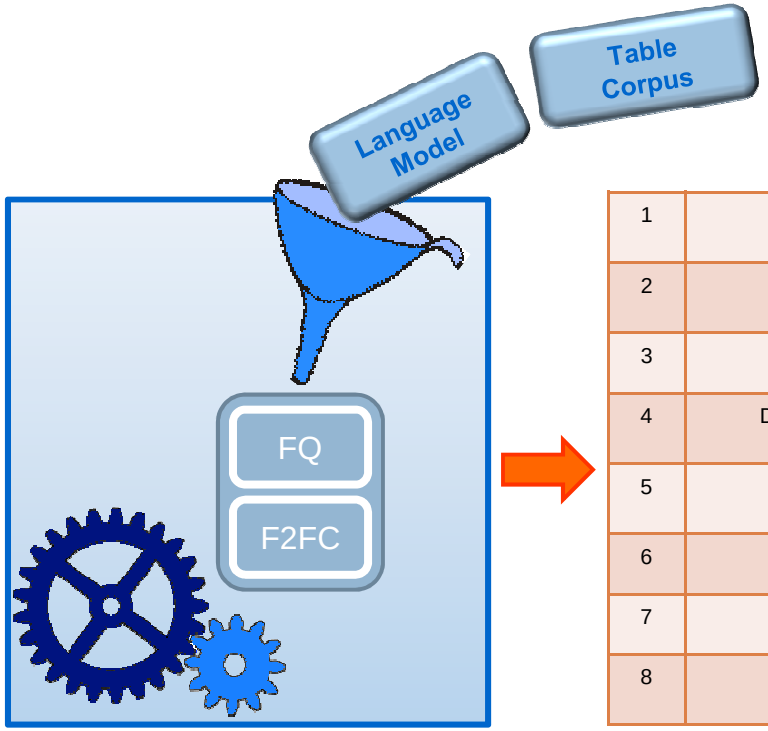
Hazem Elmeleegy Jayant Madhavan Alon Halevy
[EMH09]

Gruppo 17: Juri Castellani Emma Coradduzza Daniela Moschetto

Sommario

Introduzione

COSA SI VUOLE FARE



1	What's Opera Doc	Warnes Bros	1957
2	Duck Amuck	Warner Bros	1953
3	The Band Concert	Disney	1935
4	Duck Dodgers in the 24 1/2th Century	Warner Bros	1953
5	One Froggy Evening	Warner Bros	1956
6	Gertie The Dinosaur	McCay	
7	Red Hot Riding Hood	MGM	1943
8	Porky in Wackyland	Warner Bros	1938

Liste nel Web

Algoritmo ListExtract

Tabelle relazionali

L'algoritmo ListExtract effettua lo split delle righe della lista in campi della tabella relazionale.
Usa come risorse esterne Language Model e Table Corpus per calcolare delle funzioni di score:
Field Quality Score e Field-to-Field Consistency Score

Introduzione
COME LO SI VUOLE FARE

ListExtract è pensato per:

- operare in modo **non-supervisionato**

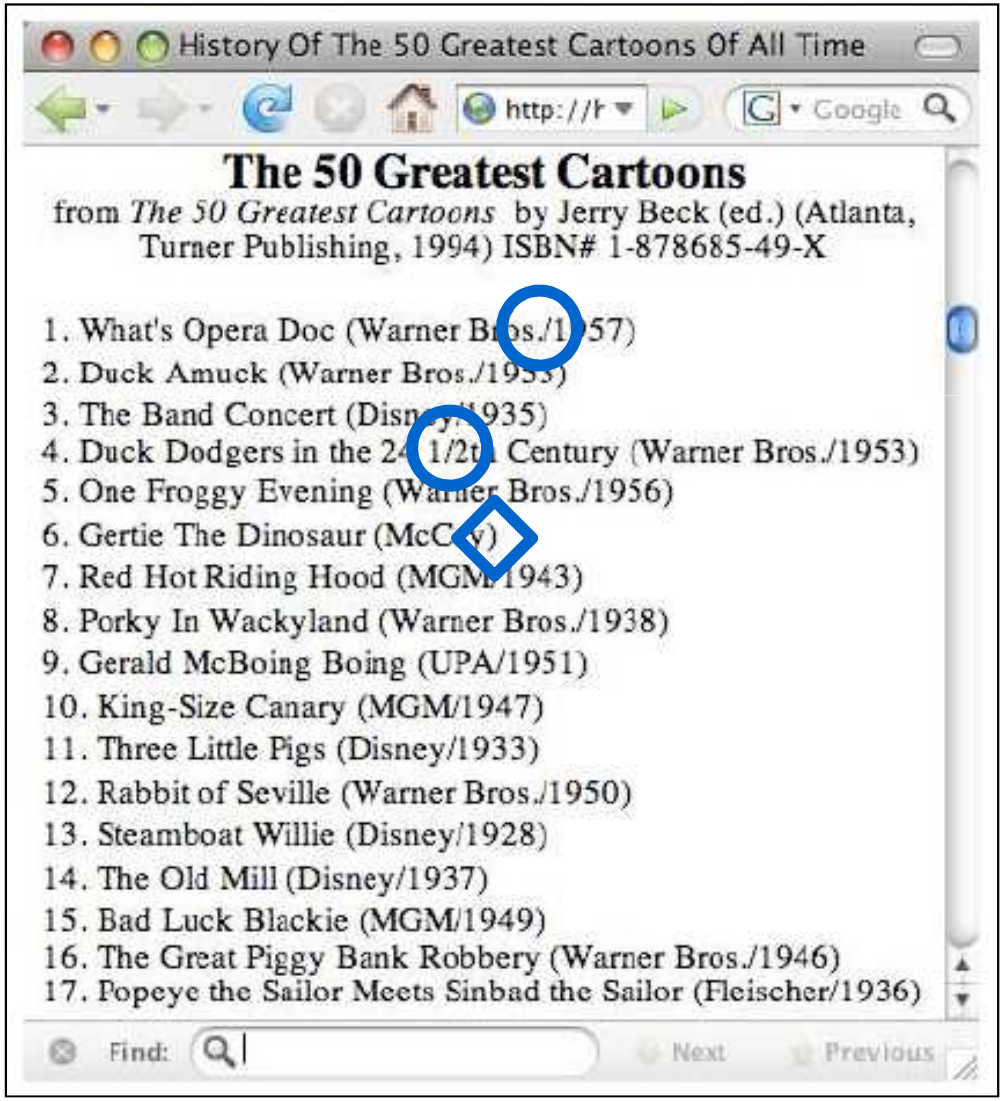
- essere **indipendente dal dominio**

Queste qualità sono essenziali per rendere la tecnica applicabile su scala web.

Introduzione
PROBLEMATICHE

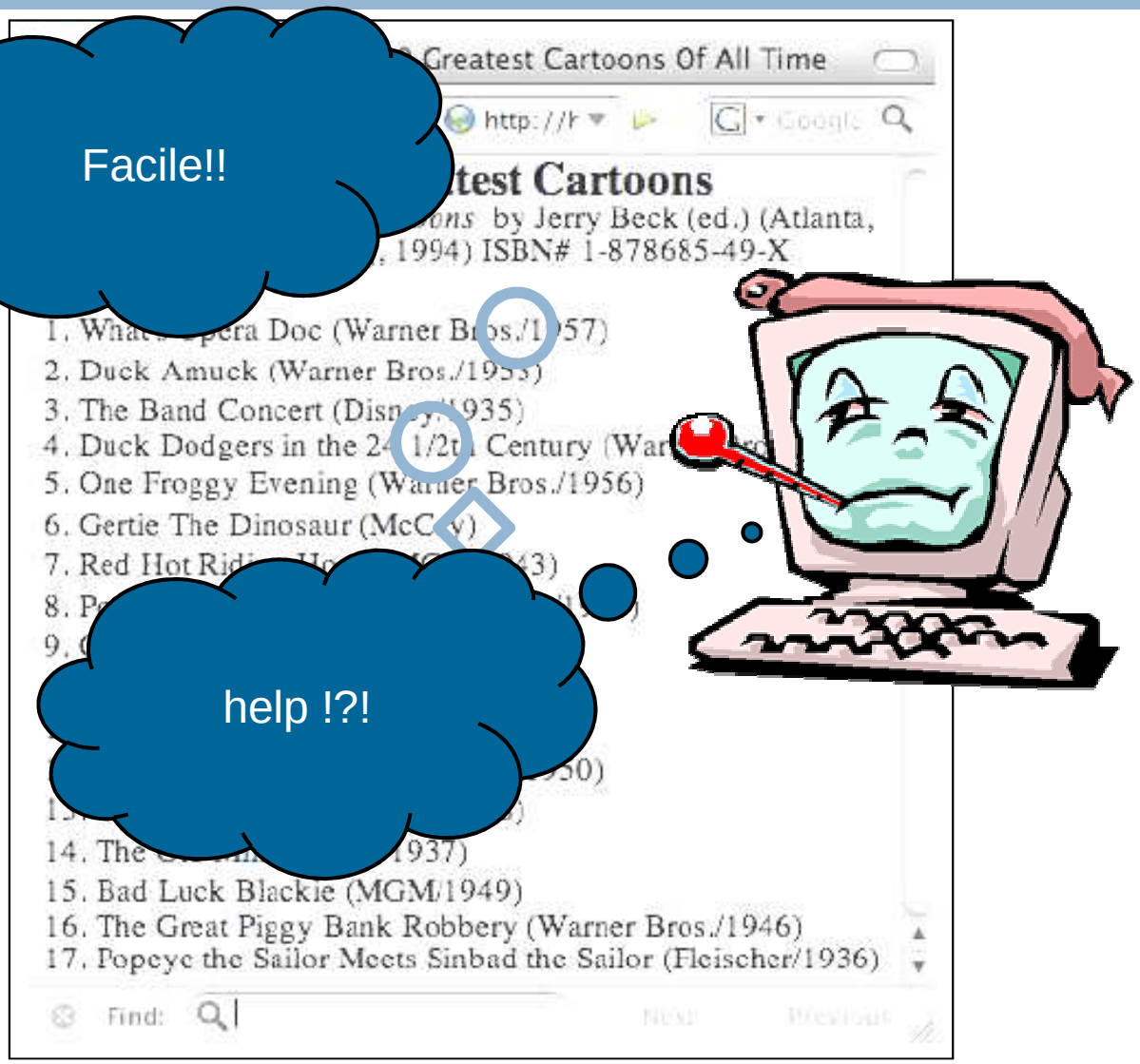
Un carattere delimitatore in alcuni casi può essere usato come delimitatore, in altri come carattere facente parte del campo (●).

É possibile avere alcune informazioni mancati (◆).



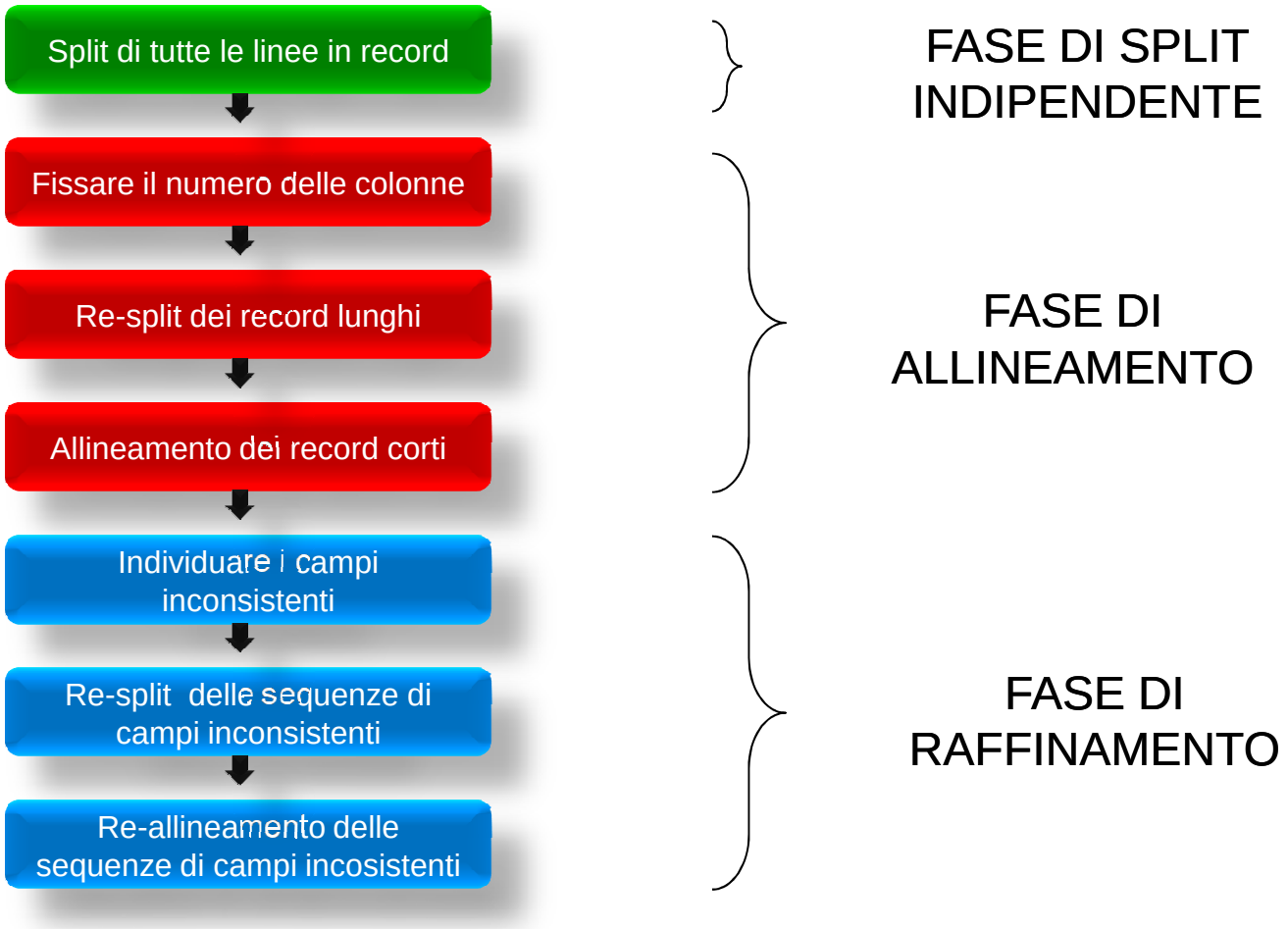
Introduzione
PROBLEMATICHE

Un carattere delimitatore
alcuni casi può essere



Sommario

Algoritmo ListExtract



Panoramica
(FASE DI SPLIT INDIPENDENTE)

1 || What's Opera Doc || Warner Bros || 1957
2 || Duck Amuck || Warner Bros || 1953
3 || The Band Concert || Disney || 1935
4. Duck Dodgers in the 24 1/2th Century (Warner Bros || 1953
5 || One Froggy Evening || Warner Bros || 1956
6 || Gertie the Dinosaur || McCay
...
17 || Popeye the Sailor || Meets || Sinbad the Sailor || Fletcher || 1936

Ogni linea della lista di input è divisa in più campi.
I record possono non avere lo stesso numero di campi.

Panoramica
(FASE DI ALLINEAMENTO)

1 || What's Opera Doc || Warner Bros || 1957
2 || Duck Amuck || Warner Bros || 1953
3 || The Band Concert || Disney || 1935
4. Duck Dodgers in the 24 1/2th Century (Warner Bros || 1953
5 || One Froggy Evening || Warner Bros || 1956
6 || Gertie the Dinosaur || McCay
...
17. Popeye the Sailor Meets || Sinbad the Sailor || Fletcher || 1936

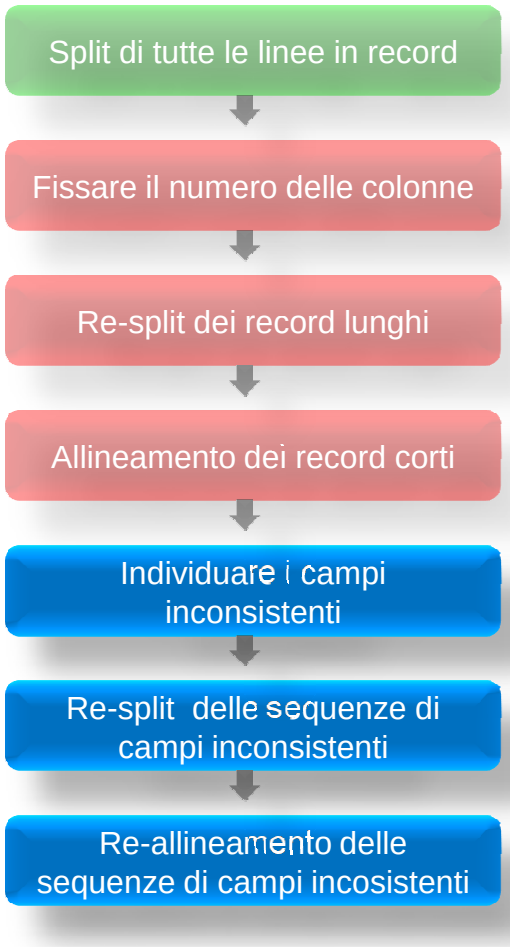
Numero colonne = 4.
Re-merge e re-split dei record lunghi.

Panoramica
(FASE DI ALLINEAMENTO)

1	What's Opera Doc	Warner Bros	1957
2	Duck Amuck	Warner Bros	1953
3	The Band Concert	Disney	1935
		4. Duck Dodgers in the 24 1/2th Century (Warner Bros	1953
5	One Froggy Evening	Warner Bros	1956
6	Gertie the Dinosaur	McCay	
...	...	...	...
17. Popeye the Sailor Meets	Sinbad the Sailor	Fletcher	1936

Otteniamo una tabella.

Panoramica
(FASE DI RAFFINAMENTO)



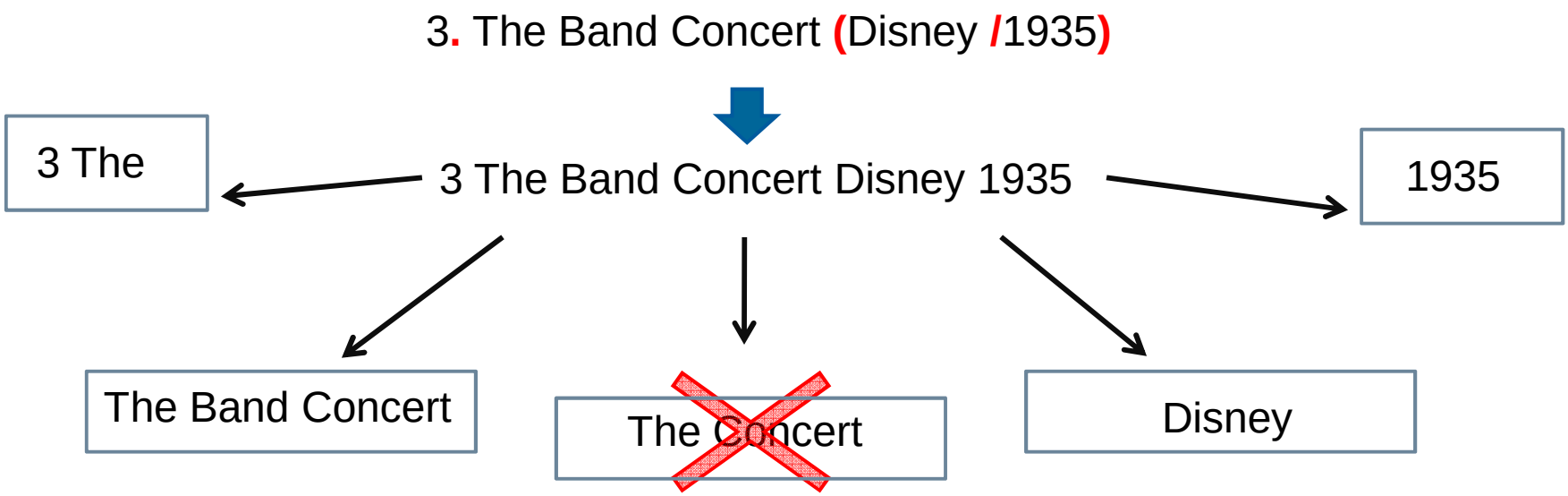
1	What's Opera Doc	Warner Bros	1957
2	Duck Amuck	Warner Bros	1953
3	The Band Concert	Disney	1935
4	Duck Dodgers in the 24 1/2th Century	Warner Bros	1953
5	One Froggy Evening	Warner Bros	1956
6	Gertie the Dinosaur	McCay	
...	...	...	...
17	Popeye the Sailor	Meets Sinbad the Sailor (Fletcher	1936

Individuazione e correzione dei campi incoerenti.

Fase di split indipendente
PREPROCESSING

Da ogni linea della lista vengono tolti i delimitatori.

Si determinano tutti i possibili campi candidati, ovvero sequenze di una o più parole consecutive.



Fase di split indipendente
FIELD QUALITY SCORE

Per misurare la qualità di un particolare campo candidato f si usa il **Field Quality Score** FQ(f).

Calcoliamo FQ come combinazione lineare di score parziali:

- **Type Score:** espressioni regolari per riconoscere vari tipi di dati ricorrenti (es. date, email, numeri telefonici, URL, ecc.)
  score: 1 se viene riconosciuto un tipo, 0 altrimenti.

- **Language Model Score:** indica la probabilità che una o più parole in sequenza hanno di formare un campo.
  score: tiene conto di quanto sia probabile che le parole di un campo candidato siano nello stesso campo e di quanto sia improbabile che stiano insieme alle parole immediatamente adiacenti (prima e dopo) al campo candidato considerato ([0..1]).

- **Table Corpus Support Score:** indica se il numero di occorrenze del campo candidato nel Table Corpus supera una data soglia.
  score: 1 se supera la soglia, 0 altrimenti.

Fase di split indipendente

SPLITLINE

Input

3. The Band Concert (Disney /1935)

pre-processing: (rimozione delimitatori)

3 The Band Concert Disney 1935

Output

	FQ Score
The Band Concert	0.92
The Band	0.89
Disney	0.82
Band Concert	0.65
Disney 1935	0.51
1935	0.34
...	...
3	0.15
Band Concert Disney	0.12
3 The Band	0.07
Concert Disney 1935	0.03
3 The Band Concert Disney 1935	0.01

Fase di split indipendente
SPLITLINE

Input

3. The Band Concert (Disney /1935)

pre-processing: (rimozione delimitatori)

3 The Band Concert Disney 1935

Output

	The Band Concert	

Subsequence	FQ Score
The Band Concert	0.92
~~The Band~~	~~0.89~~
Disney	0.82
~~Band Concert~~	~~0.65~~
Disney 1935	0.51
1935	0.34
...	...
3	0.15
~~Band Concert Disney~~	~~0.12~~
~~3 The Band~~	~~0.07~~
~~Concert Disney 1935~~	~~0.03~~
~~3 The Band Concert Disney 1935~~	~~0.01~~

Fase di split indipendente
SPLITLINE

Input

3. The Band Concert (Disney /1935)

pre-processing: (rimozione delimitatori)

3 The Band Concert Disney 1935

Output

	The Band Concert	

	The Band Concert	Disney	

Subsequence	FQ Score
The Band Concert	0.92
~~The Band~~	~~0.89~~
Disney	0.82
~~Band Concert~~	~~0.65~~
~~Disney 1935~~	~~0.51~~
1935	0.34
...	...
3	0.15
~~Band Concert Disney~~	~~0.12~~
~~3 The Band~~	~~0.07~~
~~Concert Disney 1935~~	~~0.03~~
~~3 The Band Concert Disney 1935~~	~~0.01~~

Fase di split indipendente

SPLITLINE

Input

3. The Band Concert (Disney /1935)

pre-processing: (rimozione delimitatori)

3 The Band Concert Disney 1935

Output

	The Band Concert	

	The Band Concert	Disney	

3	The Band Concert	Disney	1935

Subsequence	FQ Score	
The Band Concert	0.92	ok
~~The Band~~	~~0.89~~	
Disney	0.82	ok
~~Band Concert~~	~~0.65~~	
~~Disney 1935~~	~~0.51~~	
1935	0.34	ok
...	...	
3	0.15	ok
~~Band Concert Disney~~	~~0.12~~	
~~3 The Band~~	~~0.07~~	
~~Concert Disney 1935~~	~~0.03~~	
~~3 The Band Concert Disney 1935~~	~~0.01~~	

Fase di split indipendente
ALGORITMO SPLITLINE

Algoritmo 1 SplitLine (l: linea)

1: r = {} //insieme dei campi in cui viene splittata la linea l
2: estrarre tutte le sottosequenze da l come campi candidati.
3: calcolare FQ per ciascun campo candidato.
4: C_f = campi candidati ordinati in ordine decrescente di FQ.
5: **while** C_f non è vuota **do**
6: rimuovere f_{top}, il campo candidato con la FQ più alto in C_f.
7: aggiungere f_{top} a r.
8: rimuovere i campi candidati sovrapposti con f_{top} da C_f.
9: **end while**
10: return r

Fase di allineamento

Prima di costruire la tabella occorre **stabilire il numero di colonne k**, visto che al termine dello split indipendente i record possono presentare un numero diverso di campi.

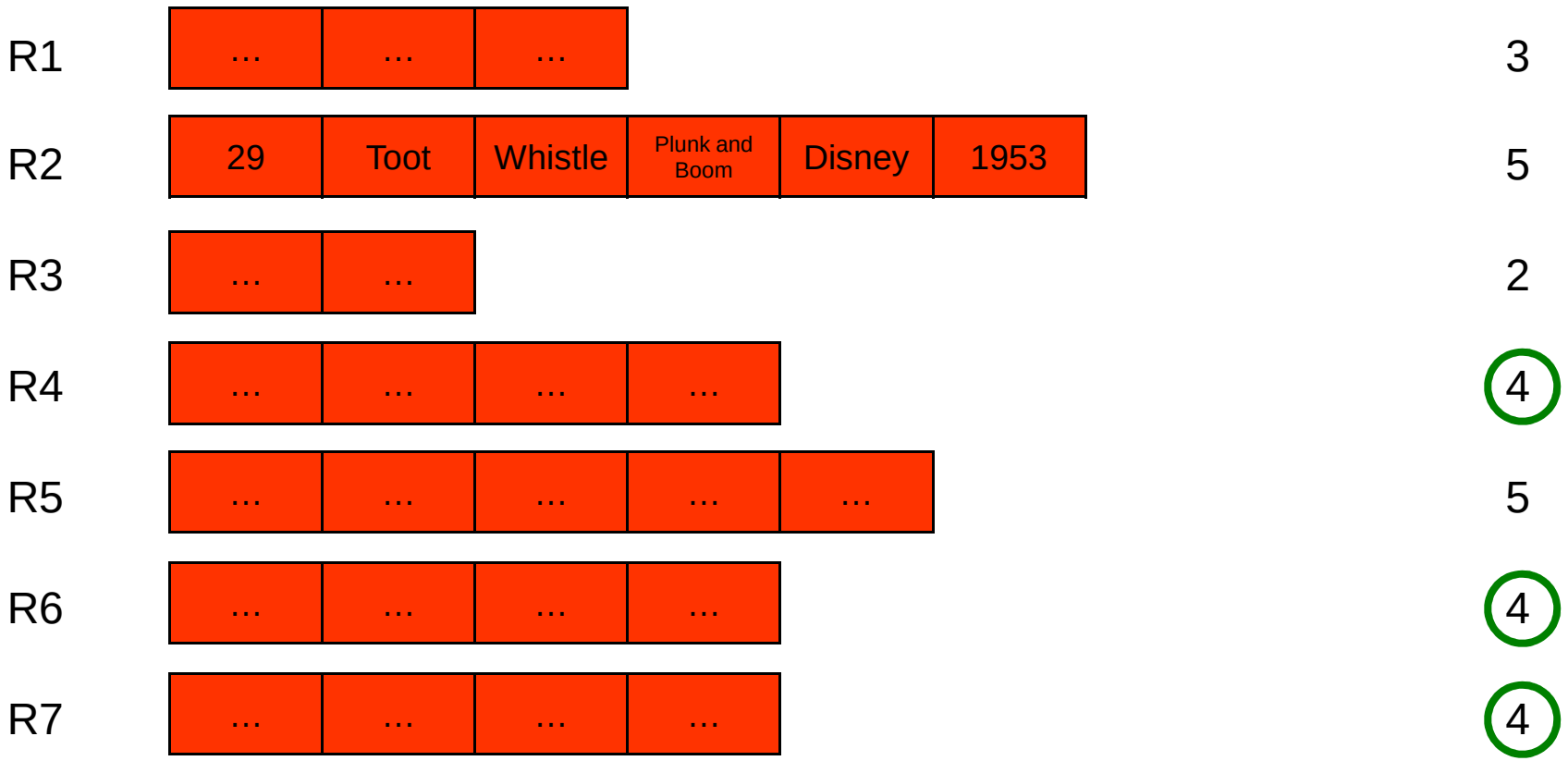
Supponendo che la prima fase divide ogni riga i-esima in k_i campi, scegliamo come valore per k il k_i più frequente nella lista.

In base al valore di k_i possiamo avere tre casi:

- **k_i = k**: allineamento banale.
- **k_i > k** (record lunghi): occorre diminuire il numero di campi affinchè il record non ne abbia più di k, usiamo **BoundedSplitLine** (variante di SplitLine).
- **k_i < k** (record corti): occorre inserire i campi “NULL” nei record corti, usiamo **AlignShortRecord**.

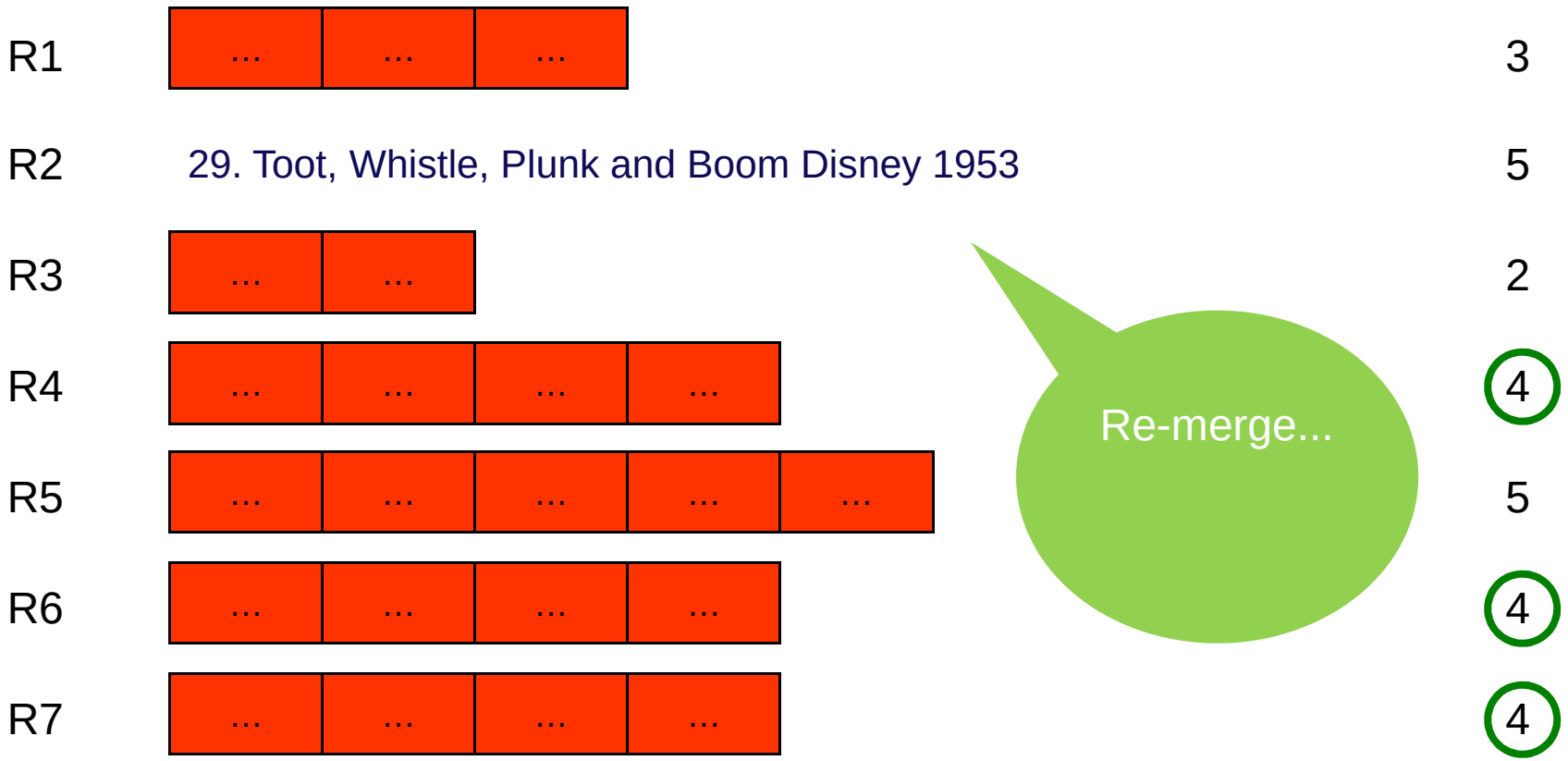
Fase di allineamento
BOUNDEDSPLITLINE

Il valore più frequente tra i k_i è 4 → k = 4.
La tabella finale avrà 4 colonne!!!
Eseguiamo il BoundedSplitLine...



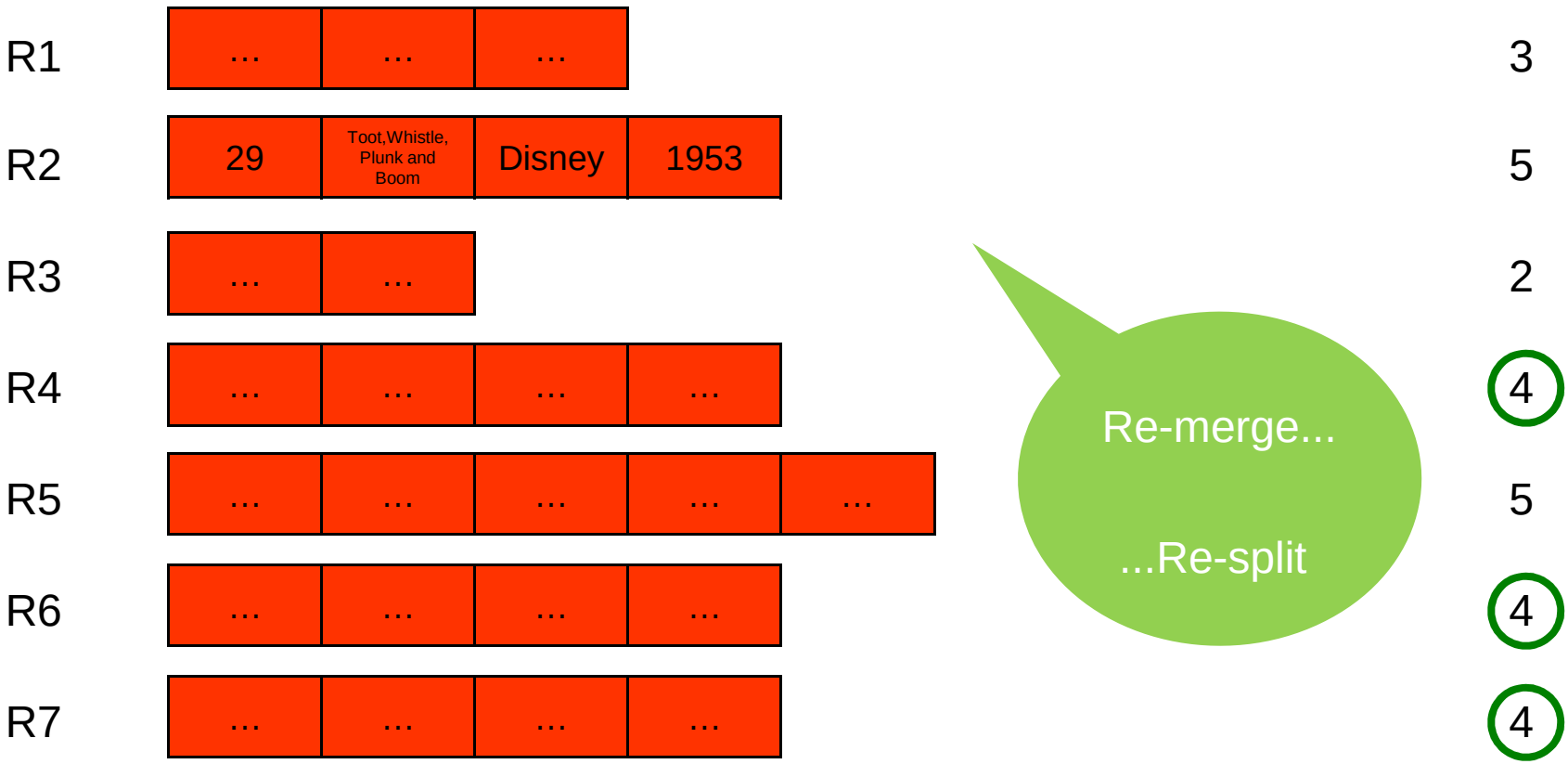
Fase di allineamento
BOUNDEDSPLITLINE

Il valore più frequente tra i k_i è 4 → k = 4.
La tabella finale avrà 4 colonne!!!
Eseguiamo il BoundedSplitLine...



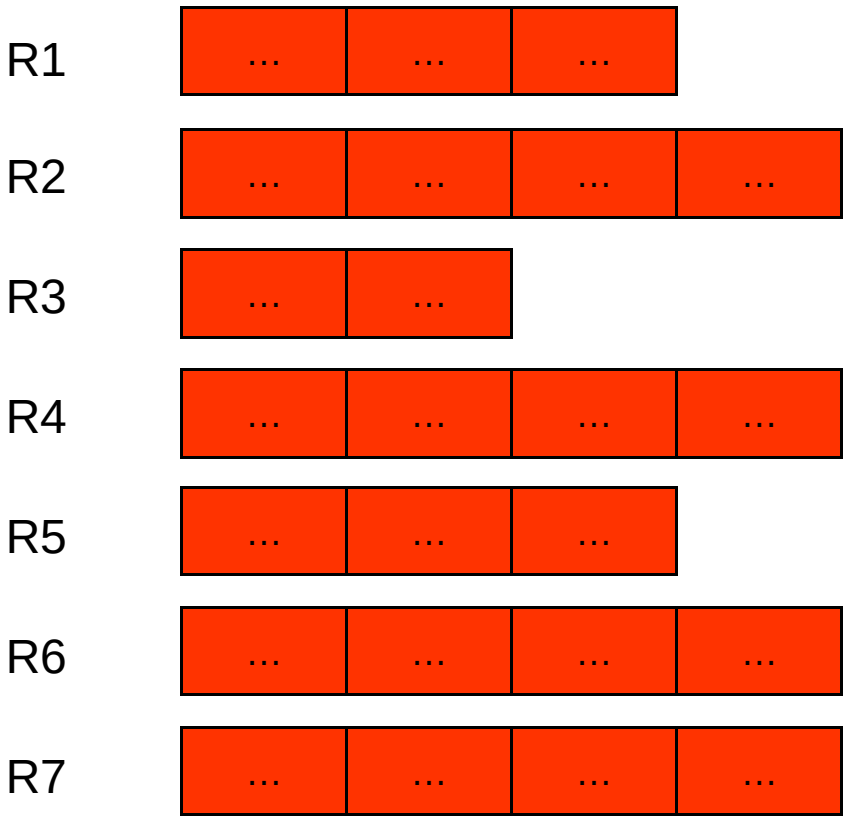
Fase di allineamento
BOUNDEDSPLITLINE

Il valore più frequente tra i k_i è 4 → k = 4.
La tabella finale avrà 4 colonne!!!
Eseguiamo il BoundedSplitLine...



Fase di allineamento
BOUNDEDSPLITLINE

... Il risultato del BoundedSplitLine è il seguente:



Il numero dei campi di R2 ed R5 ora è ≤ k.

Fase di allineamento
ALGORITMO BOUNDEDSPLITLINE

Algoritmo 2 BoundedSplitLine (l:linea, k_{max}:limite superiore)

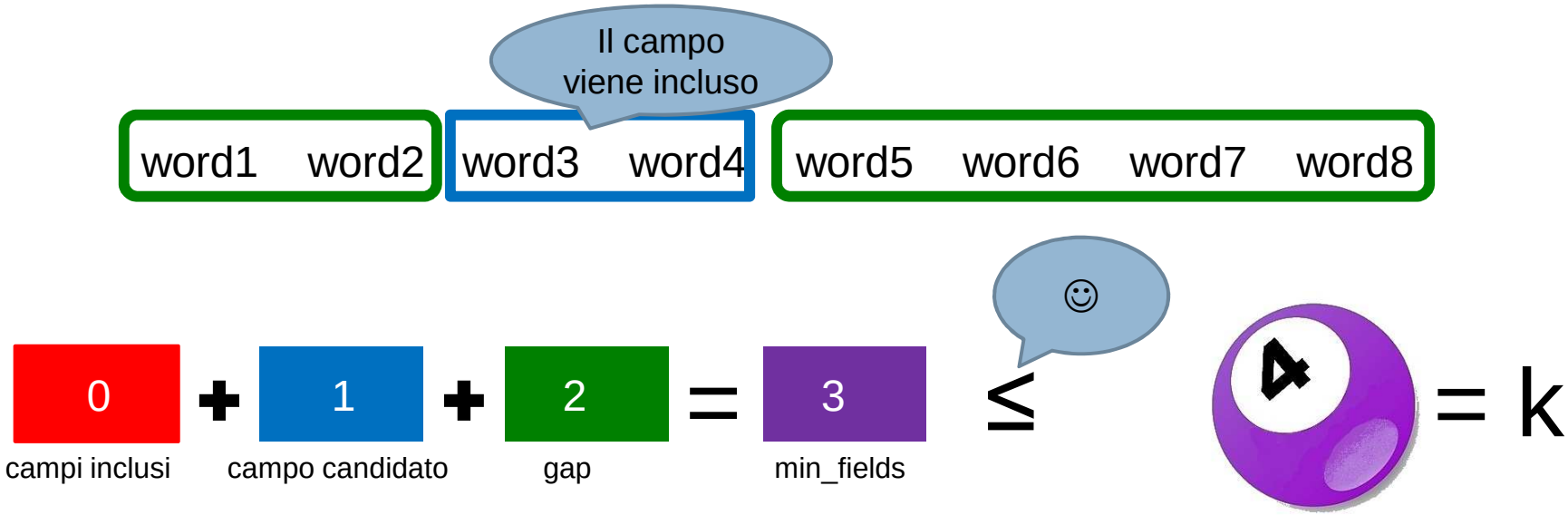
1: r = {} //insieme dei campi in cui viene splittata la linea l
2: estrarre tutte le sottosequenze da l come campi candidati.
3: calcolare FQ per ciascun campo candidato.
4: C_f= campo candidato ordinato in ordine decrescente di FQ.
5: **while** C_f non è vuota **do**
6: rimuovere f_{top}, il campo candidato con la FQ più alta in C_f.
7: stima min_fields(r, f_{top}),cioè il numero minimo di campi se f_{top} è stato incluso in r.
8: **if** min_fields(r,f_{top}) ≤ k_{max} **then**
9: aggiungi f_{top} a r.
10: rimuovi i campi candidati sovvrapposti con f_{top} da C_f.
11: **end if**
12: **end while**
13: return r.

Fase di allineamento
FUNZIONE MIN_FIELDS

La funzione **min_fields(record, campo candidato)** calcola il numero minimo di campi che la riga in esame avrebbe se il campo candidato considerato venisse aggiunto al record associato alla riga.

Somma:

- numero di campi già inclusi nel record
- + 1, ovvero il campo candidato
- gap: numero di sequenze di parole consecutive non assegnate a campi.

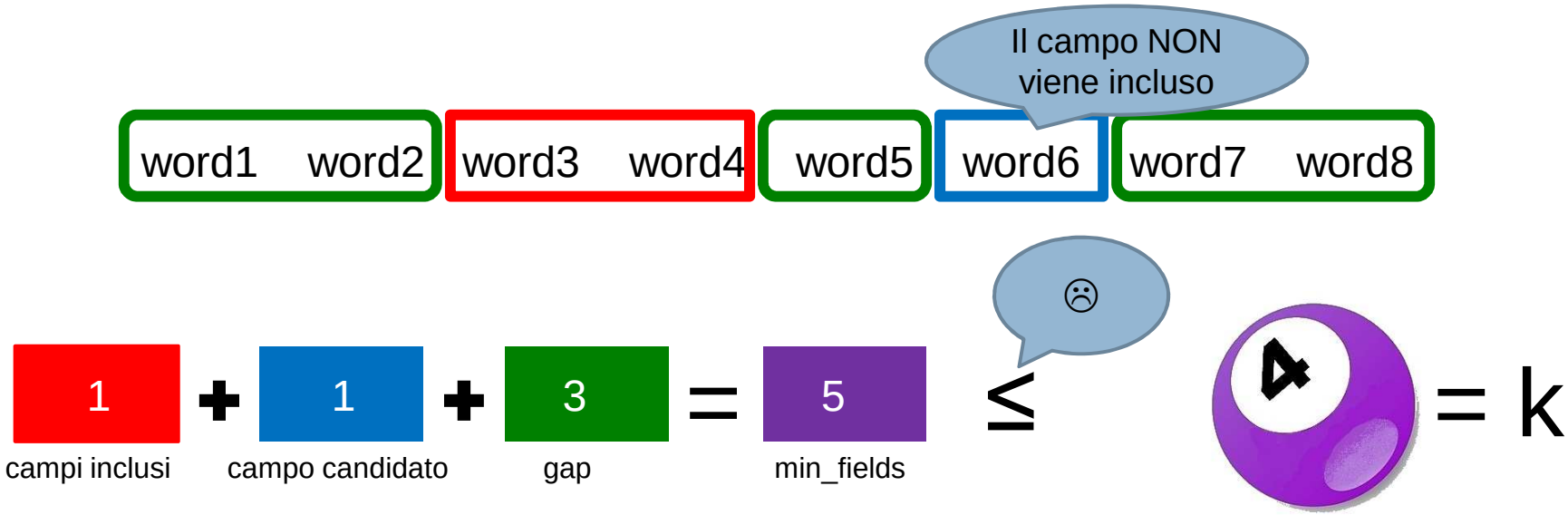


Fase di allineamento
FUNZIONE MIN_FIELDS

La funzione **min_fields(record, campo candidato)** calcola il numero minimo di campi che la riga in esame avrebbe se il campo candidato considerato venisse aggiunto al record associato alla riga.

Somma:

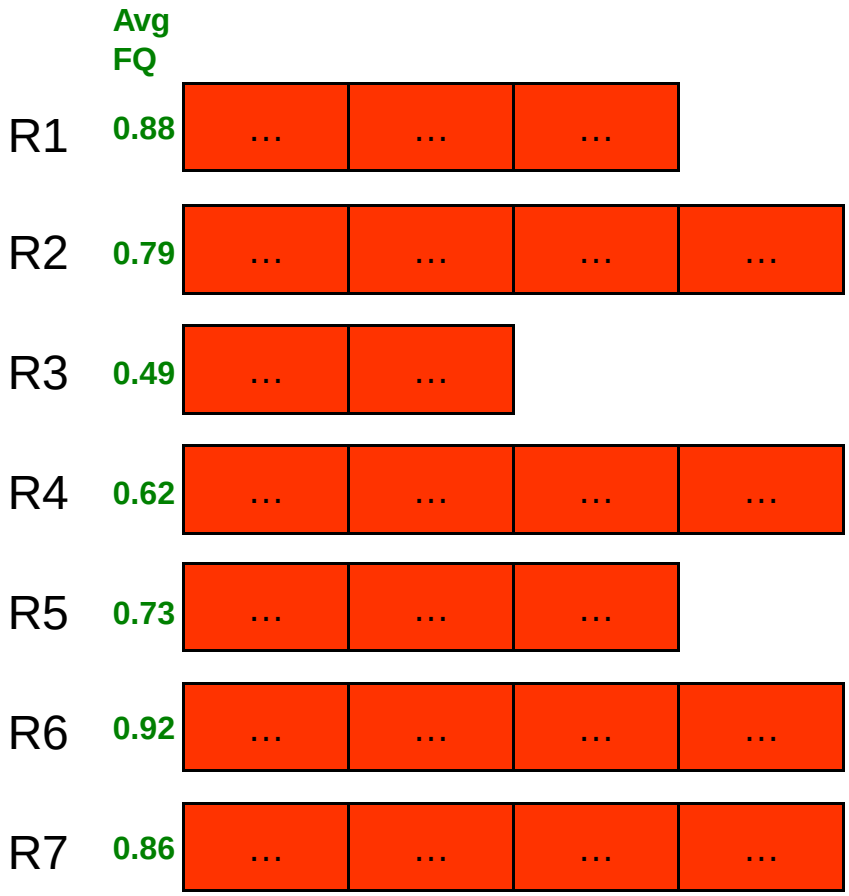
- numero di campi già inclusi nel record
- + 1, ovvero il campo candidato
- gap: numero di sequenze di parole consecutive non assegnate a campi.



Fase di allineamento

I NULL vengono inseriti nelle posizioni delle informazioni mancanti.

Ogni campo non NULL si deve allineare con la colonna che gli è più simile.



Fase di allineamento

I record vengono ordinati in base al numero di campi.

In caso di parità ci basiamo sul valore decrescente di FQ medio.

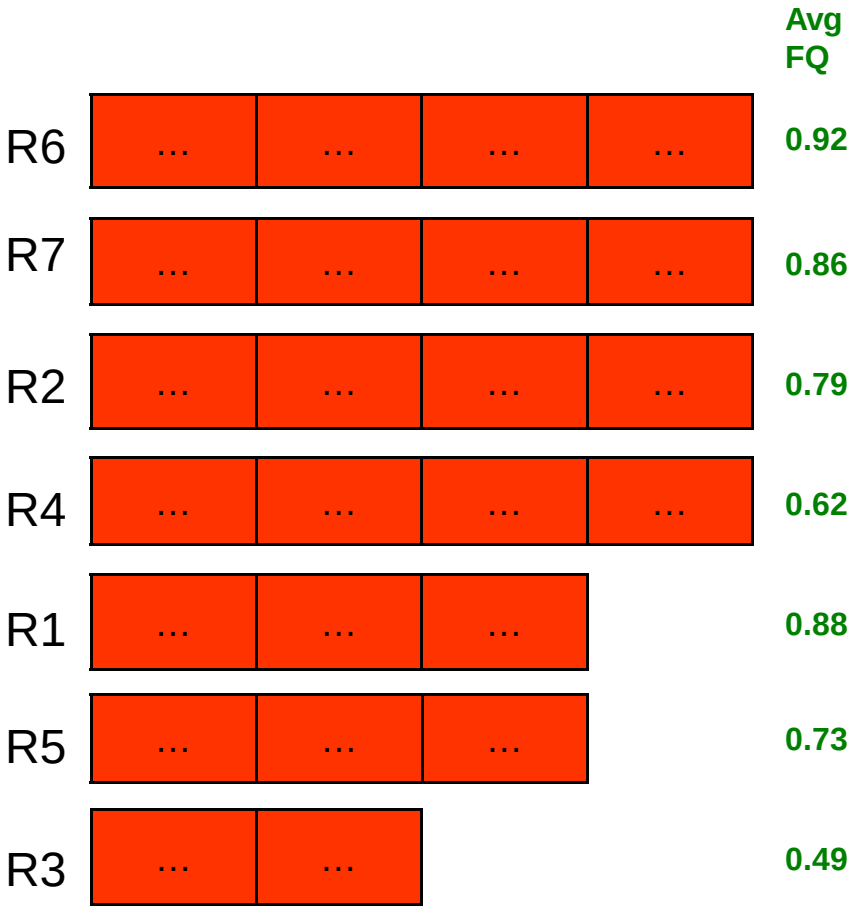
Prima dell’ordinamento.

Fase di allineamento

I record vengono ordinati in base al numero di campi.

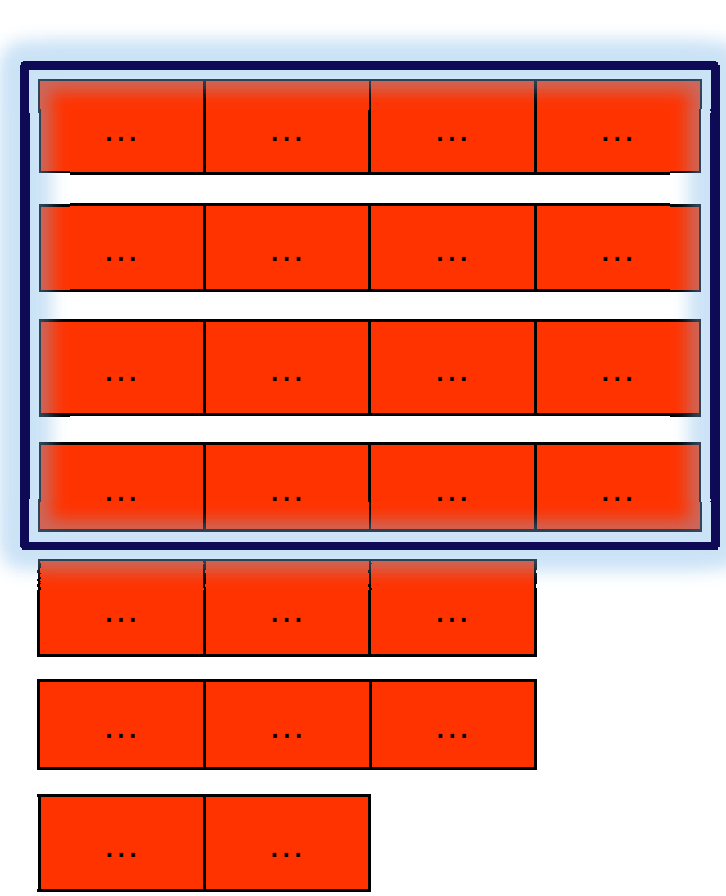
In caso di parità ci basiamo sul valore decrescente di FQ medio.

Dopo l’ordinamento.



Fase di allineamento

Otteniamo una tabella parziale con k = 4 colonne, associandovi 1 a 1 i campi delle righe che hanno k campi.



Fase di allineamento

Per ogni riga con k_i<k campi usiamo l'algoritmo AlignShortRecord.

In corrispondenza delle informazioni mancanti vengono inseriti i NULL.

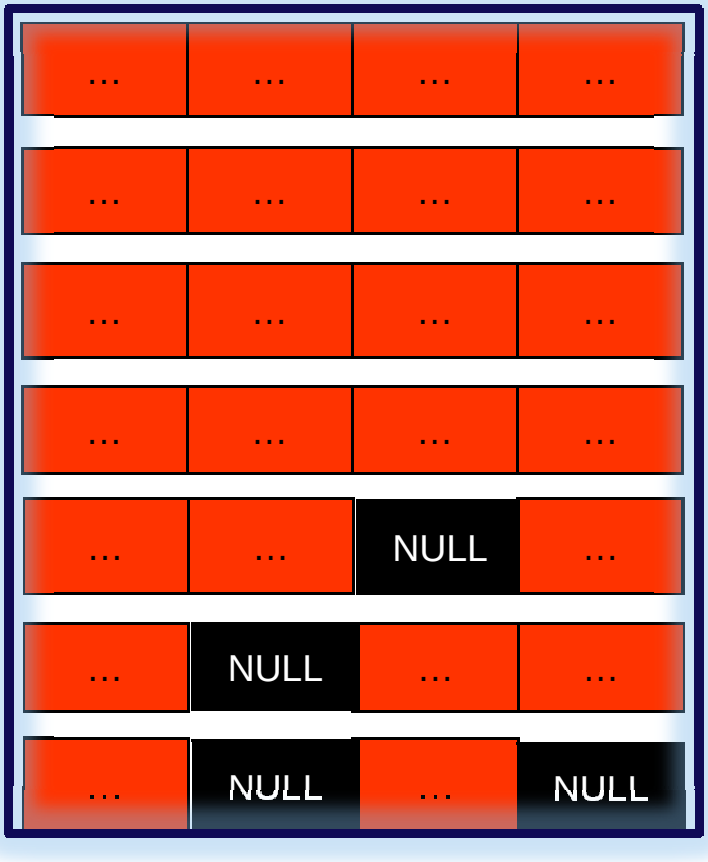


Tabella parziale

Avg FQ
0.92
0.86
0.79
0.62
0.88
0.73
0.49

Fase di allineamento

Per ogni riga con meno di k campi otteniamo l'allineamento creando una matrice di costi M (k^- x k, dove k^- è il numero di campi nella riga considerata).

Ogni elemento M[i,j] rappresenta il costo del migliore allineamento del campo f_i della riga in esame con la colonna c_j della tabella parziale.

Definiamo tre funzioni di costo:

Matched(f_i, c_j) = F2FC(f_i, c_j)

- Matched(f_i,c_j) = costo associato al fatto che il campo f_i sia allineato con la colonna c_j.

NULL nel campo c_j.
UnMatched(c_j) = C, costante.

- UnMatched(c_j) = costo associato al fatto che nessun campo della riga i sia allineato con la colonna c_j.

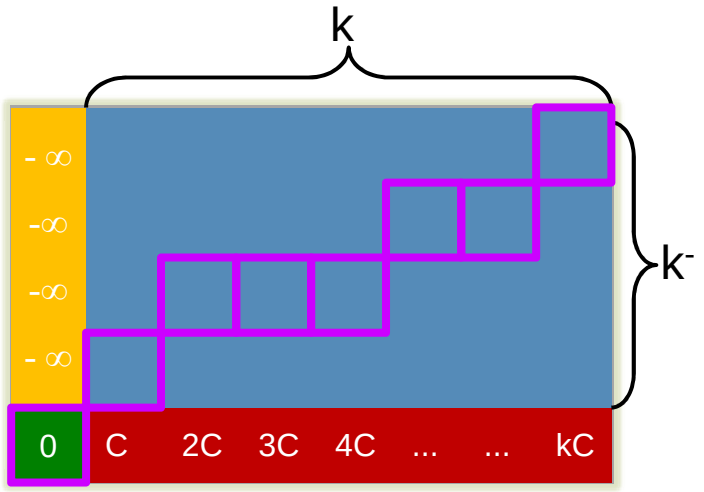
Ogni campo deve SEMPRE fare match con una colonna → UnMatched(f_i) = -∞

- UnMatched(f_i) = costo associato al fatto che nessuna colonna della tabella parziale è allineata con il campo f_i.

Fase di allineamento
ALGORITMO ALIGNSHORTRECORD

Algoritmo 3 AlignShortRecord (r: record con k^- campi, T:Tabella parziale con k colonne)

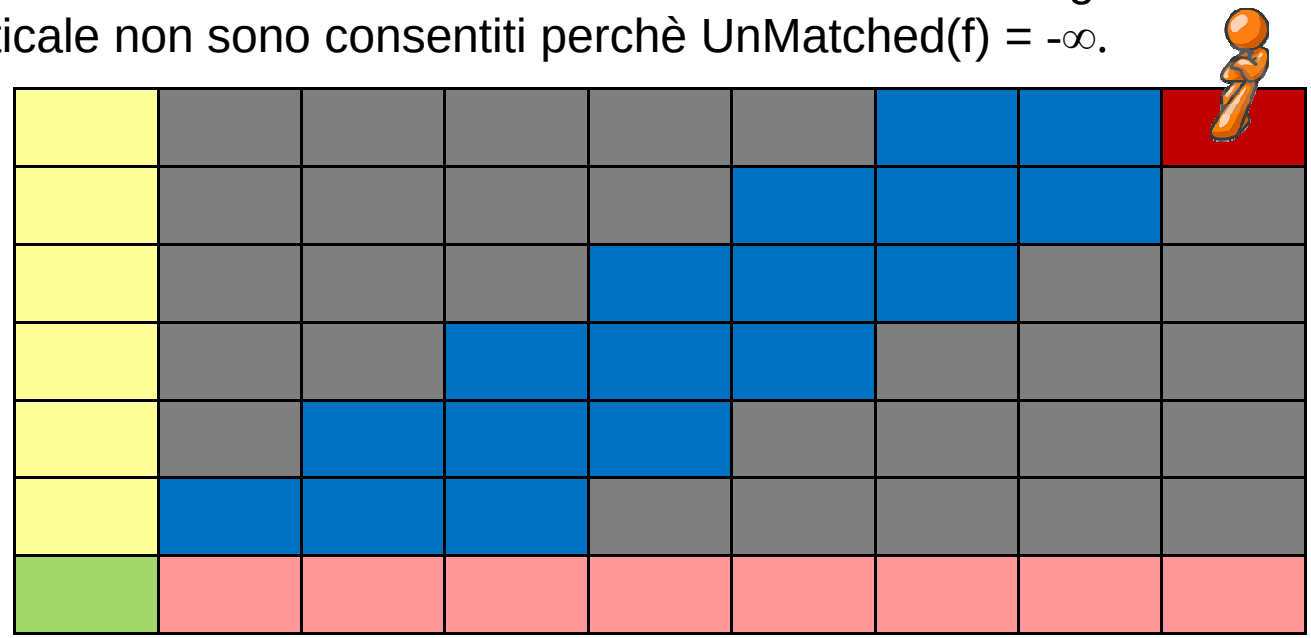
1: M[0, 0] = 0
2: **for** i = 1 a k^- **do**
3: M[i, 0] = M[i - 1, 0] + UnMatched(f_i)
4: **end for**
5: **for** j = 1 to k **do**
6: M[0, j] = M[0,j - 1] + UnMatched (c_j)
7: **end for**
8: **for** i = 1 a k^- **do**
9: **for** j = 1 a k **do**
10: M[i, j] = max { M[i, j-1] + UnMatched (c_j), M[i-1, j] + UnMatched (f_i), M[i-1, j-1] + Matched (f_i, c_j) }
11: **end for**
12: **end for**
13: ritorna il miglior allineamento A[k^-, k] partendo da M[k^-, k] tornando a M[0, 0].



Matrice dei costi del record r

Fase di allineamento

Intuitivamente un allineamento corretto può stare SOLO all'interno di un "corridoio di celle".
Gli unici movimenti consentiti sono in orizzontale e in diagonale.
In verticale non sono consentiti perchè UnMatched(f) = -∞.



Se così non fosse ci sarebbero dei campi non allineati con nessun colonna.

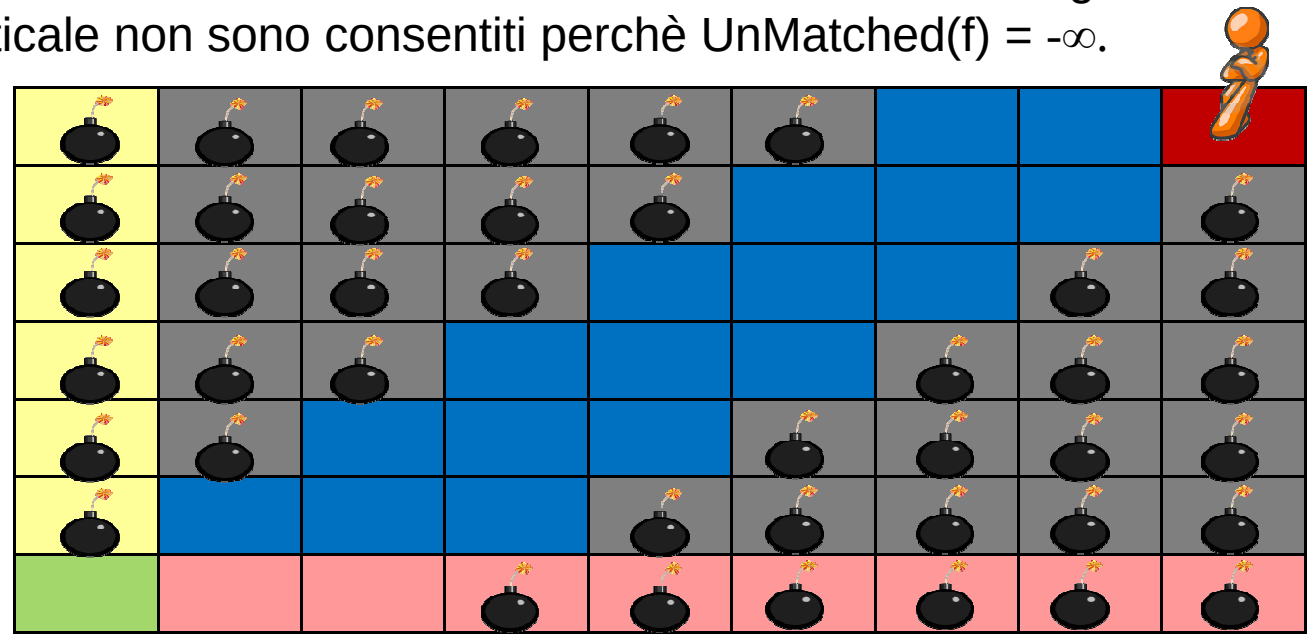
Fase di allineamento

Intuitivamente un allineamento corretto può stare SOLO all'interno di un "corridoio di celle".
Gli unici movimenti consentiti sono in orizzontale e in diagonale.
In verticale non sono consentiti perchè UnMatched(f) = -∞.

Se così non fosse ci sarebbero dei campi non allineati con nessun colonna.

Fase di allineamento

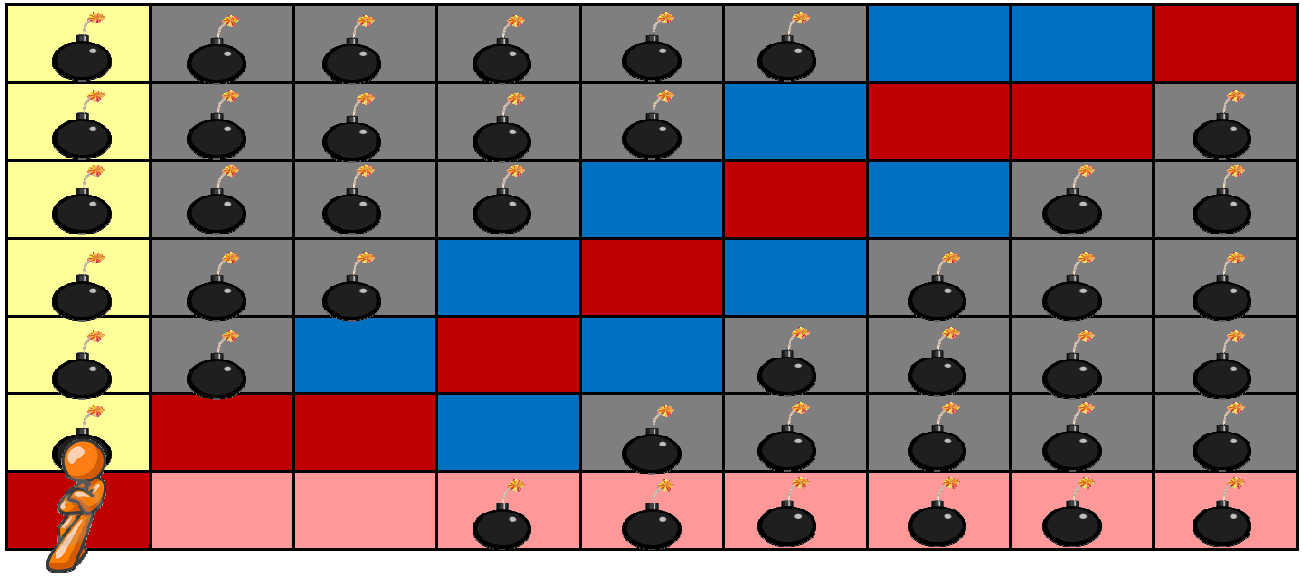
Intuitivamente un allineamento corretto può stare SOLO all'interno di un "corridoio di celle".
Gli unici movimenti consentiti sono in orizzontale e in diagonale.
In verticale non sono consentiti perchè UnMatched(f) = -∞.



Se così non fosse ci sarebbero dei campi non allineati con nessun colonna.

Fase di allineamento

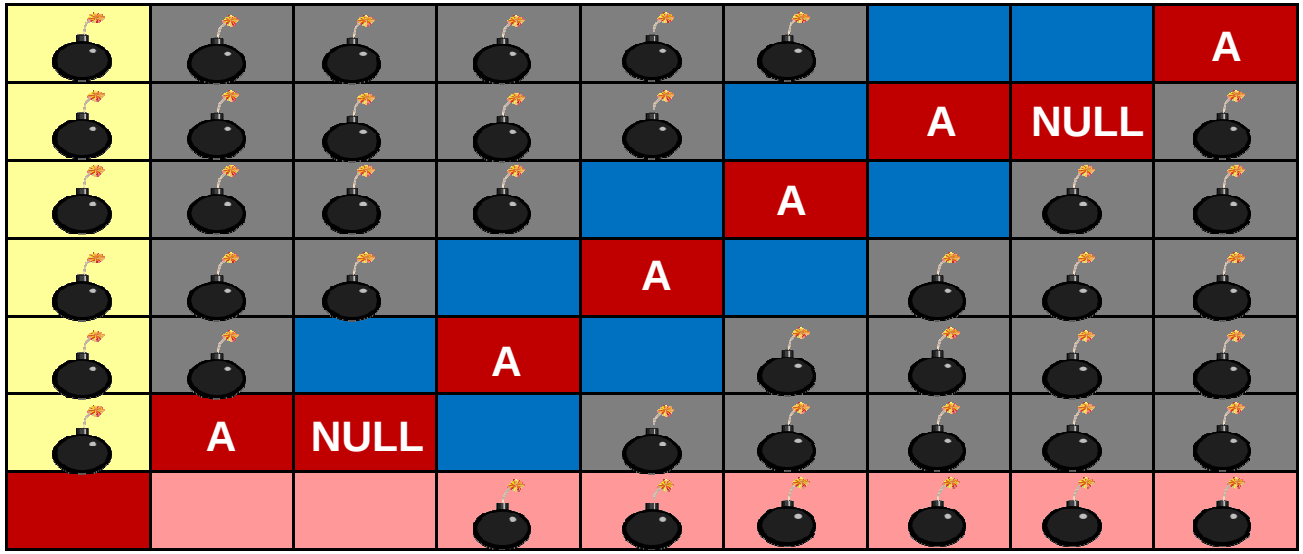
Intuitivamente un allineamento corretto può stare SOLO all'interno di un "corridoio di celle".
Gli unici movimenti consentiti sono in orizzontale e in diagonale.
In verticale non sono consentiti perchè UnMatched(f) = -∞.



Se così non fosse ci sarebbero dei campi non allineati con nessun colonna.

Fase di allineamento

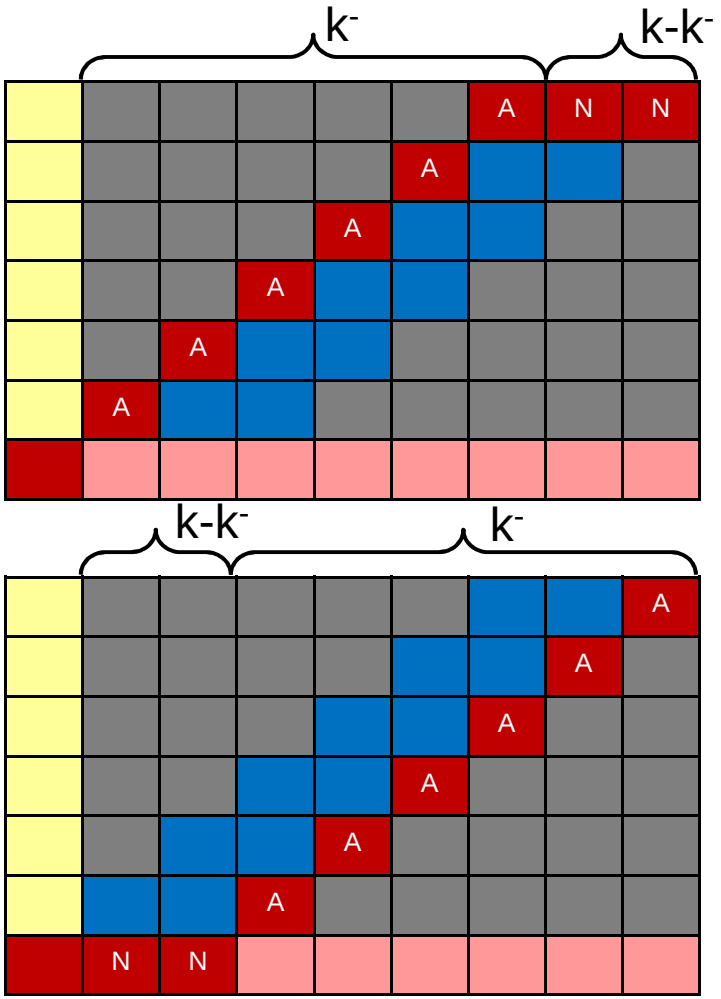
Intuitivamente un allineamento corretto può stare SOLO all'interno di un "corridoio di celle".
Gli unici movimenti consentiti sono in orizzontale e in diagonale.
In verticale non sono consentiti perchè UnMatched(f) = -∞.



Se così non fosse ci sarebbero dei campi non allineati con nessun colonna.

Fase di allineamento

Individuiamo due casi limite:



k- campi della riga in esame sono allineati con le prime k- colonne della tabella parziale e le restanti k-k- sono NULL

k- campi della riga in esame sono allineati con le ultime k- colonne della tabella parziale e i primi k-k- sono a NULL

Fase di allineamento
FIELD TO FIELD CONSISTENCY SCORE

La funzione usata per calcolare il costo associato all'allineamento del campo f con la colonna c, rappresenta quanto bene f si allinea con c.
È usata per calcolare Matched(f,c):

$$\text{Matched}(f,c) = \text{F2FC}(f,c) = 1/n \times \sum_{i=1..n} \text{F2FC}(f,f_i^c)$$

Da ora in poi al posto di $\text{F2FC}(f,f_i^c)$ scriveremo $\text{F2FC}(f_1,f_2)$.
$\text{F2FC}(f_1,f_2)$ misura la similarità fra una coppia di campi candidati f_1 e f_2.
Si calcola come combinazione lineare di score parziali:

$$\text{F2FC}(f_1,f_2) = a_{tc} \times S_{tc}(f_1,f_2) + a_{tcc} \times S_{tcc}(f_1,f_2) + a_{sc} \times S_{sc}(f_1,f_2) + a_{dc} \times S_{dc}(f_1,f_2)$$

- **Type Consistency Score**: misura l'uguaglianza tra tipi ricorrenti
  score: 1 se f_1 e f_2 sono dello stesso tipo, 0 altrimenti.

- **Tables Corpus Consistency Score**: valore alto se f_1 e f_2 nel table corpus ricorrono spesso nella stessa colonna.
  score: media tra le probabilità condizionali $\Pr(f_1|f_2)$ e $\Pr(f_2|f_1)$ ([0..1]).

Fase di allineamento
FIELD TO FIELD CONSISTENCY SCORE

- **Syntax Consistency Score**: si calcolano dei sotto-score parziali ottenuti confrontando a2a2 delle feature sintattiche (numero di lettere, percentuale di cifre, ecc) estratte dai valori dei campi.

  score: media dei sotto-score parziali normalizzati a 1 ([0..1]).

- **Delimiters Consistency Score**: misura la similarità tra i delimitatori di campo:
  " ,;:./\()<>&|ꟾ?"

  score: 1 se i delimitatori fanno match da entrambi i lati del campo

  (f_1, (f_2,

  0,5 se i delimitatori fanno match da un lato solo del campo

  (f_1, (f_2)

  0 se i delimitatori non fanno match

  ,f_1, (f_2)

Fase di allineamento
FIELD SUMMARIES

Ricordiamo che:

$$F2FC(f,c) = 1/n \times \sum_{i=1..n} F2FC(f,f_i^c)$$

$F2FC(f,f_i^c)$ viene calcolata n volte per ogni cella della matrice ($k \times k^-$), e tutto questo per ogni riga non allineata: il costo computazionale è troppo alto.

Si decide di non calcolare tutti i $F2FC(f,f_i^c)$, ma di farlo solo per dei campi ritenuti rappresentativi della colonna tra quelli già allineati: i **field summaries**.

Il loro numero è fissato come parametro: *max_n_reps*.

Sono memorizzati in una tabella di $k \times$ *max_n_reps* e aggiornati ogni volta che si aggiunge un record alla tabella parziale.

Column 1	Column 2	Column 3	Column 4
11	Steamboat Willie	Disney	1943
10	Rabbit of Seville	MGM	1935
15	Three Little Pigs	Disney	1949

I campi di una stessa riga non riguardano lo stesso record.

Fase di allineamento
ALGORITMO CREATETABLE

Algoritmo 4 CreateTable (R: lista di record)

1: **for** r_i in R **do**
2: **if** il numero di campi in r_i > k **then**
3: **AlignLongRecord**(r_i,k)
4: **end if**
5: **end for**
6: T_i = {} //matrice che contiene le righe della tabella da costruire
7: SF = {} //matrice che contiene i field summaries
8: ordina R in modo decrescente di numero di campi.
9: **for** r_i in R **do**
10: **if** il numero di campi in r_i < k **then**
11: r_i = **AlignShortRecord**(r_i, SF)
12: **end if**
13: aggiungi r_i a T_i .
14: SF = **UpdateFieldSummaries**(r_i, SF)
15: **end for**
16: ritorna T_i

Fase di raffinamento

Finora lo split è stato fatto:

- in maniera libera (fase di split indipendente),
- vincolata ad un numero massimo di colonne (fase di allineamento).

In questa fase lo si farà considerando anche i valori degli altri campi nelle colonne oltre al numero massimo di colonne.

L'obiettivo del raffinamento è trovare e correggere gli errori risultanti dalle fasi precedenti.

Si fanno due assunzioni:

- il numero dei campi corretti è molto più alto di quello dei campi scorretti.
- gli errori di split non sono isolati.

Fase di raffinamento
FQ RIVISTO

- Per misurare la qualità di un particolare campo candidato f si usa il **Field Quality Score** FQ(f).

- Calcoliamo FQ come combinazione lineare di score parziali:
  - **Type Score**
  - **Language Model Score**
  - **Table Corpus Support Score**
  - **List Support Score:** favorisce i campi candidati che sono più consistenti con le colonne interessate dagli errori.

Fase di raffinamento

- Si calcola il F2FC(f,c) di ogni campo f con la sua colonna c.
- Si ordinano in modo descrescente.

Tabella di Output

…	…	…	…	…	…
…	…	…	…	…	…
…	…	…	…	…	…
…	…	…	…	…	…
…	…	…	…	…	…
…	…	…	…	…	…

Fase di raffinamento

- Si calcola il F2FC(f,c) di ogni campo f con la sua colonna c.
- Si ordinano in modo descrescente.
- Partendo dall'ultimo si prendono come campi incoerenti quelli con il peggiore F2FC fino ad arrivare al Pinc% di tutti i campi (Pinc% parametro fissato).
- Si scartano i campi singoli e i NULL.

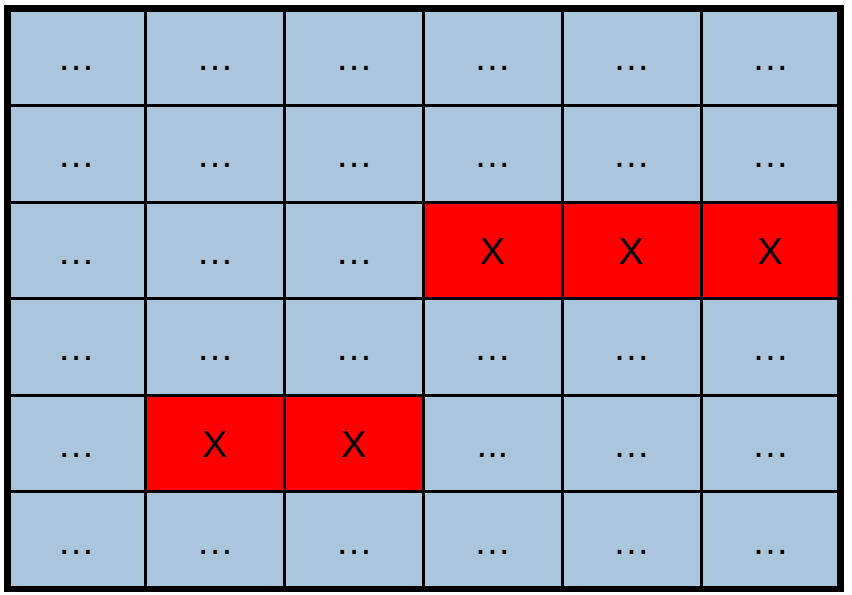
Tabella di Output

…	…	…	…	X	…
…	…	X	…	…	…
X	…	…	X	X	X
…	…	…	…	…	…
…	X	X	…	…	…
…	…	…	…	…	…

Fase di raffinamento

- Si calcola il F2FC(f,c) di ogni campo f con la sua colonna c.
- Si ordinano in modo descrescente.
- Partendo dall'ultimo si prendono come campi incoerenti quelli con il peggiore F2FC fino ad arrivare al Pinc% di tutti i campi (Pinc% parametro fissato).
- Si scartano i campi singoli e i NULL.
- Rimangono solo le sequenze di campi inconsistenti in un unico record: gli streaks.

Tabella di Output



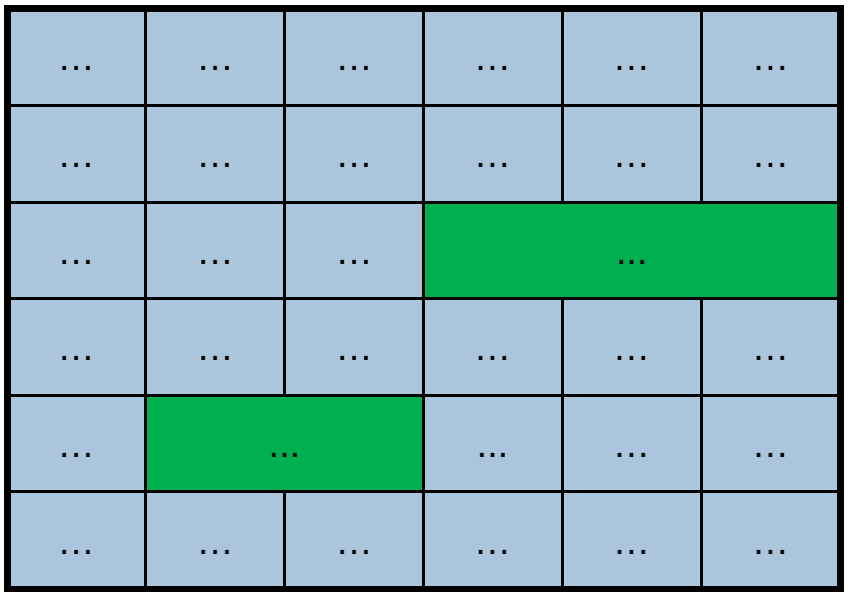
Fase di raffinamento

Assunzione #1

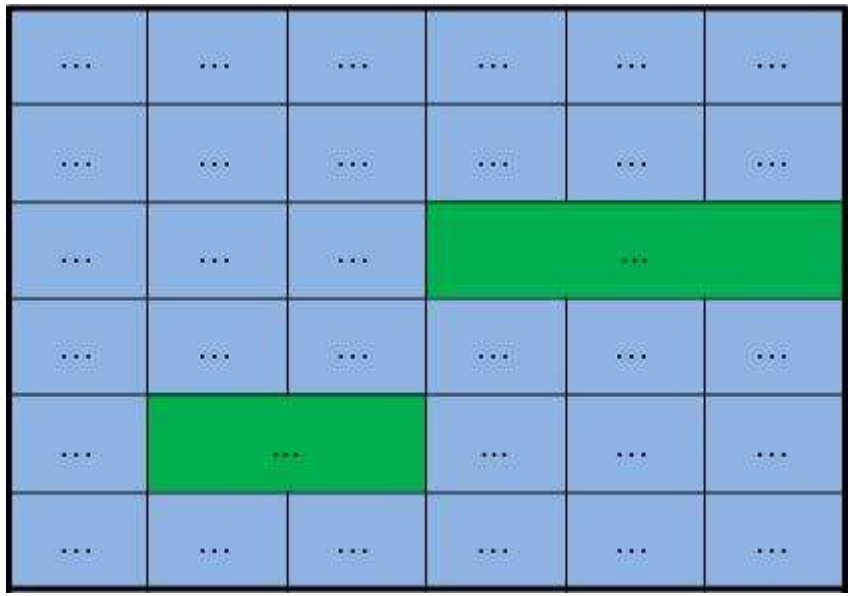
- Si calcola il F2FC(f,c) di ogni campo f con la sua colonna c.
- Si ordinano in modo descrescente.
- Partendo dall'ultimo si prendono come campi incoerenti quelli con il peggiore F2FC fino ad arrivare al Pinc% di tutti i campi (Pinc% parametro fissato).
- Si scartano i campi singoli e i NULL.
- Rimangono solo le sequenze di campi inconsistenti in un unico record: gli streaks.

Assunzione #2

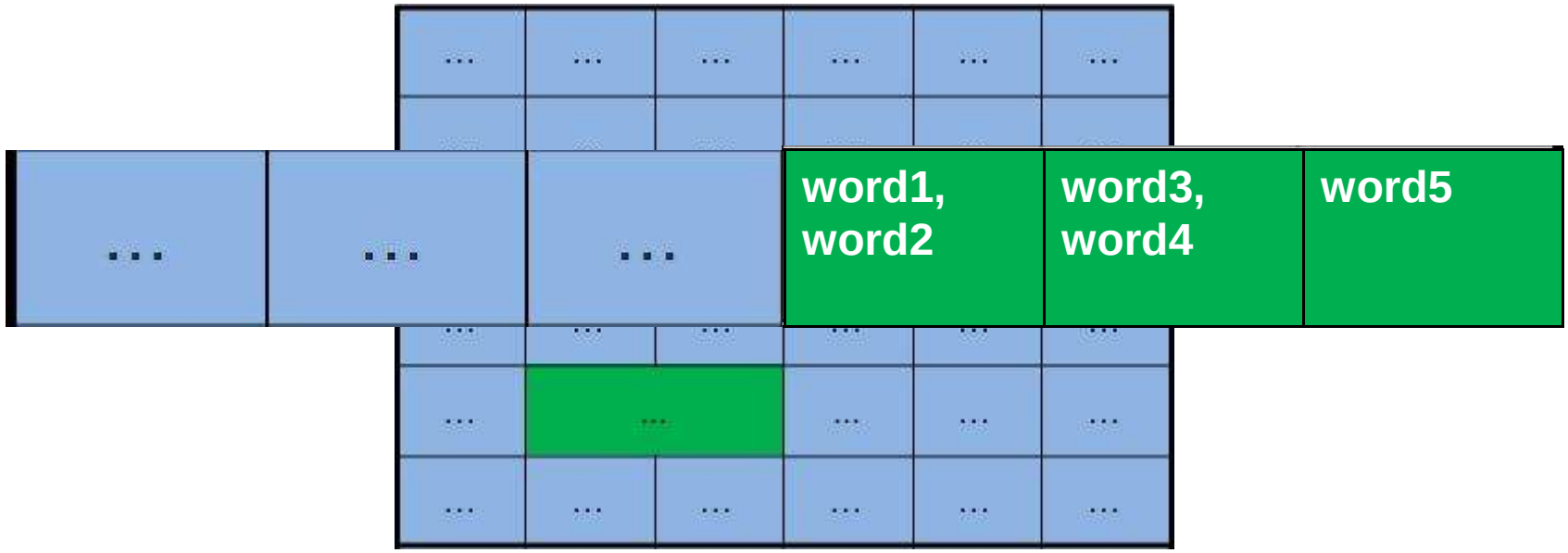
Tabella di Output



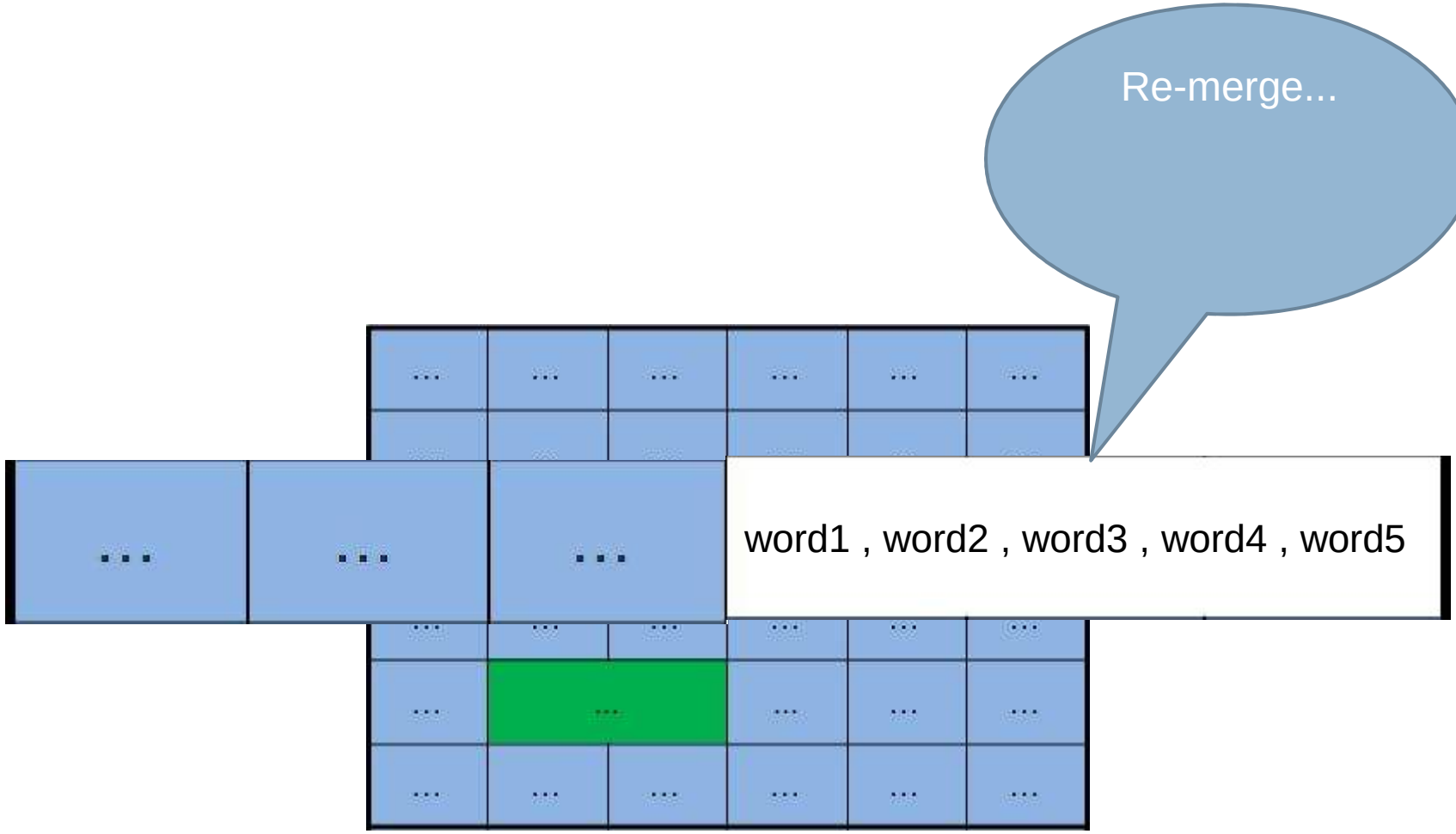
Fase di raffinamento



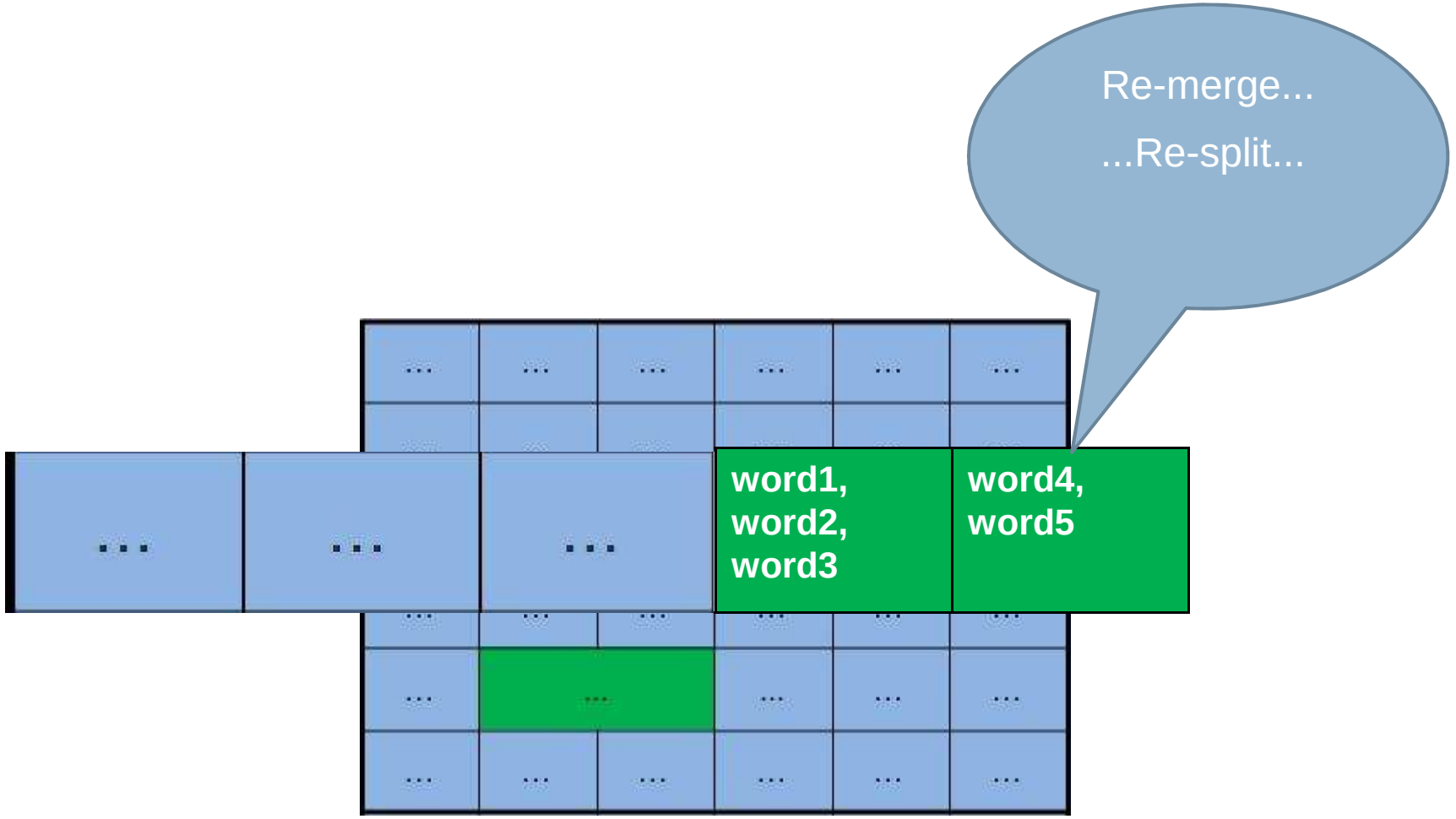
Fase di raffinamento



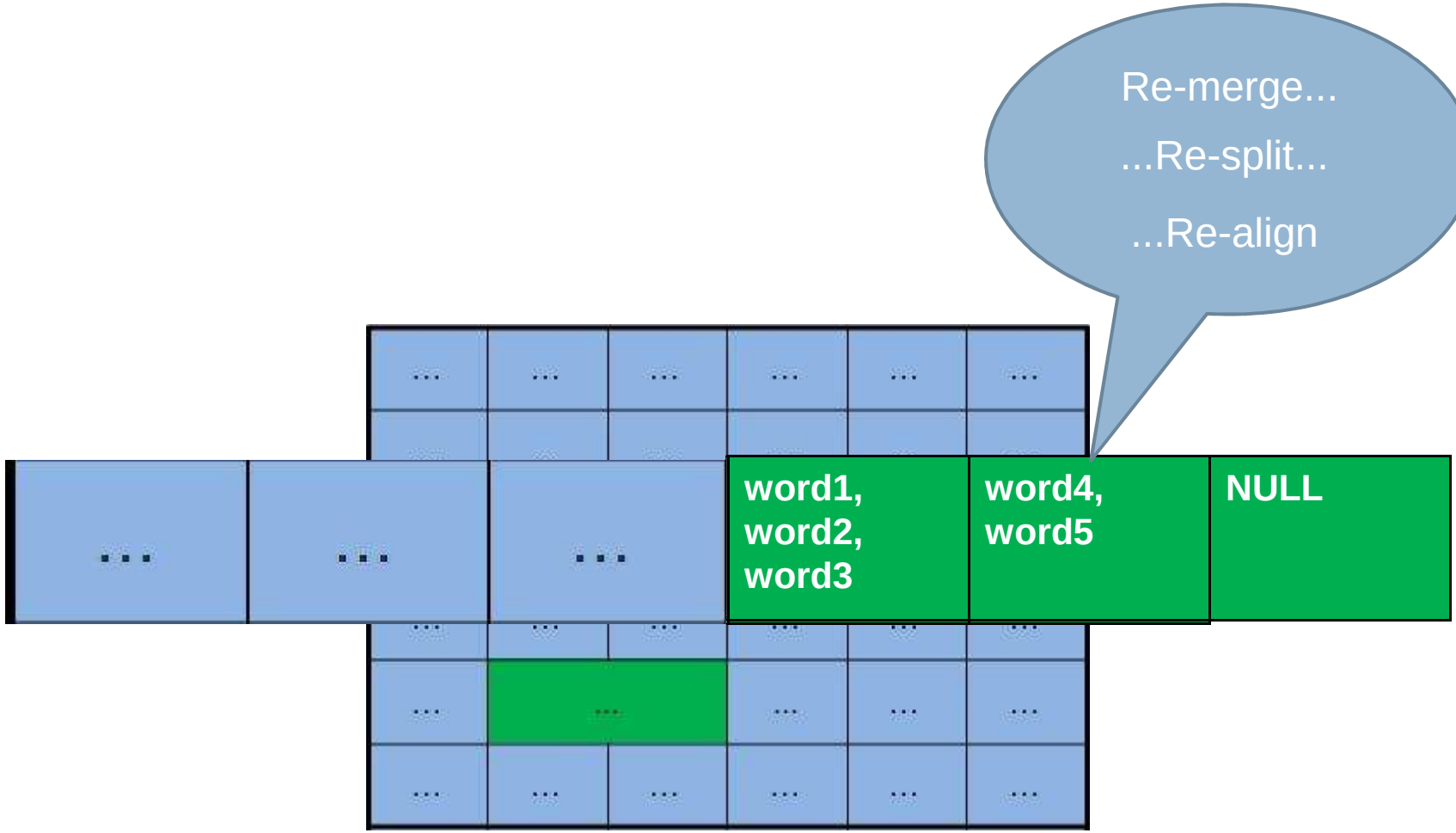
Fase di raffinamento



Fase di raffinamento



Fase di raffinamento



Sommario

Esperimenti

Per la valutazione di ListExtract vengono usati due distinti data sets, nella stessa lingua, ognuno con una propria base di verità con cui fare il confronto:

- **Tables-Derived List** (TDLists) = formato da liste estratte da tabelle, ottenute collassando tutte le celle di un record in una sola linea, con gli spazi bianchi che separano le parole.
  - Base di verità : le tabelle originarie.

- **Web List** (WLists) = formato da liste HTML ricavate dal Web e caratterizzate da domini differenti
  - Base di verità : tabelle ottenute manualmente dai creatori.

Un confronto diretto delle tabelle estratte da ListExtract e la base di verità può essere molto complesso, soprattutto perchè ci può essere più di una soluzione accettabile.

Esperimenti

Per ogni tabella estratta dalla lista, calcoliamo il **Table Extraction Score** TE(T), calcolando la media degli FQ di tutti i campi della tabella di output di ListExtract.

TE indica quanto bene la tabella è stata estratta in base alla logica dell'algoritmo (cioè in base alla FQ) e viene usato come termine di confronto per la valutazione dei risultati.

F-measure è una misura di qualità dell'estrazione; tiene conto:

- del numero di celle della tabella correttamente estratte rispetto al numero di celle totali estratte
- del numero di celle della tabella correttamente estratte rispetto al numero di celle totali presenti nella base di verità di tale tabella.

Esperimenti

COSTRUZIONE DEI GRAFICI

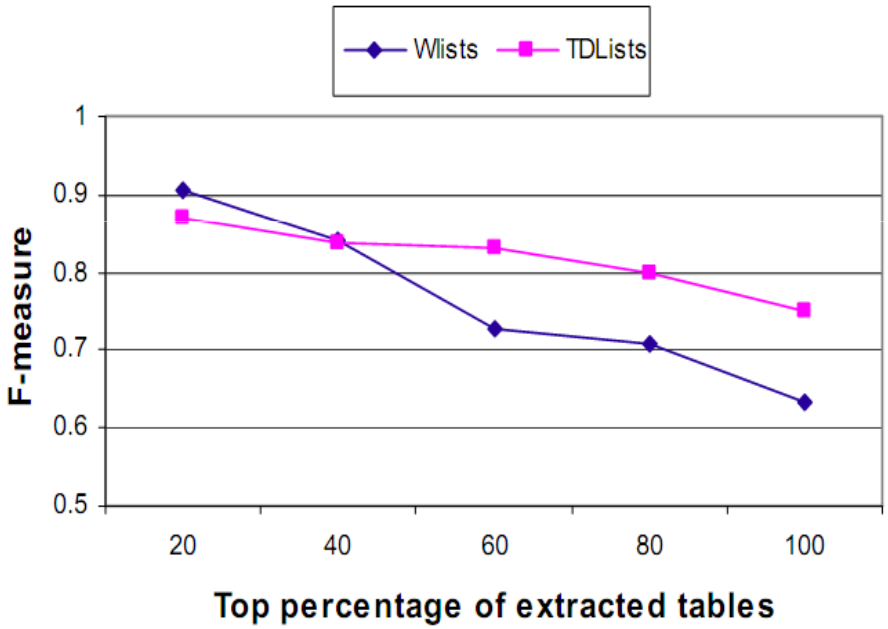
Per ogni data sets si calcolano tutti i TE delle tabelle estratte e li si ordinano in senso decrescente.

L'**ascissa** di tutti i grafici rappresenta la percentuale delle prime tabelle in tale ordine.

L'**ordinata** rappresenta il valore medio di f-measure.

Un punto <x,y> della curva indica che le prime x% tabelle ordinate secondo TE hanno un f-measure media pari a y.

Esperimenti
PERFORMANCE GLOBALI

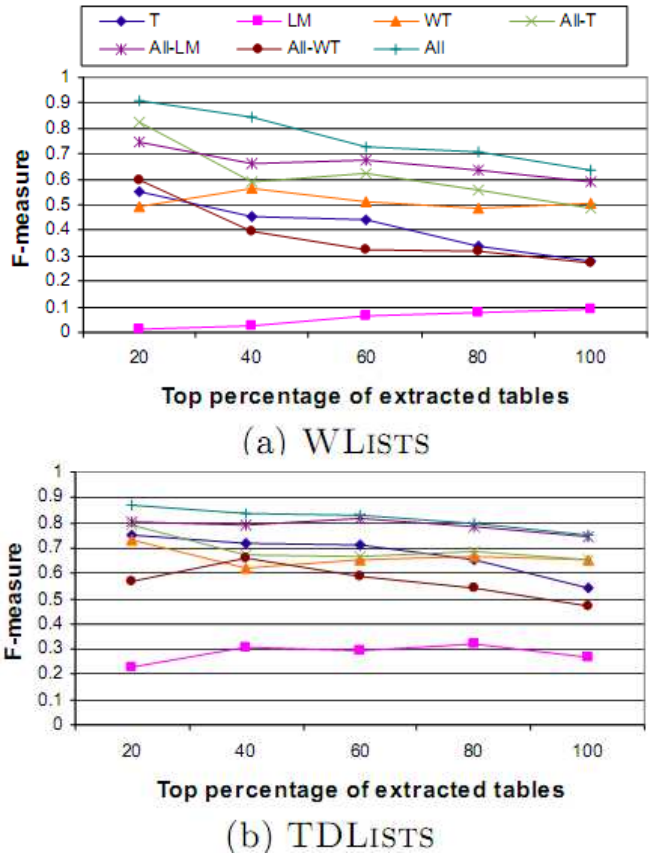


La f-measure diminuisce man mano che si considerano tabelle con TE più basso.

Le f-measure ottenute sul TDLists sono maggiori rispetto a quelle ottenute su WLists.

Le prestazioni sul TDLists sono migliori perchè le liste sottostanti sono sempre ottenute da tabelle relazionali.

Esperimenti
PERFORMANCE FQ



Si considerano 3 delle 4 componenti della FQ, tranne la componente List Support Score che è indispensabile nella fase di raffinamento.

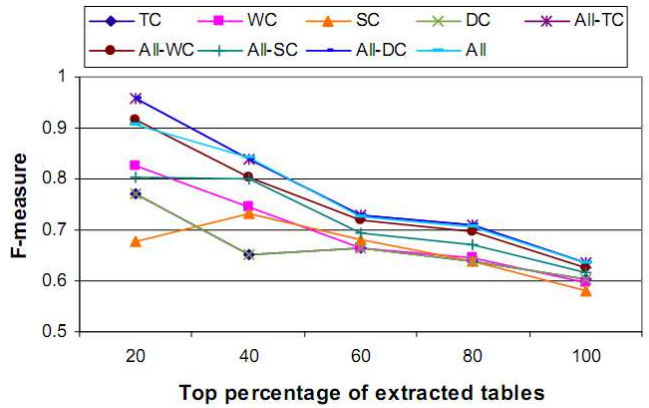
I grafici della f-measure mostrati considerano:
- tutte e tre le componenti assieme (All)
- le singole componenti separate (T , LM, WT)
- tutte le componenti escudendone di volta in volta una (All-T, All-LM, All-WT).

WT ha valori maggiori di T e LM
All-T e All-LM hanno valori maggiori di All-WT → WT aiuta ad individuare molto bene un campo.

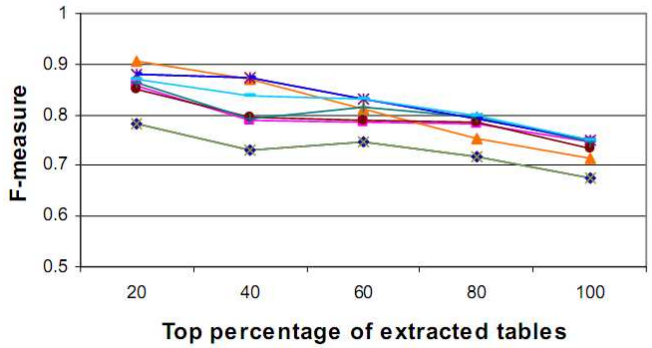
LM ha valori bassi (<20%), ma All-LM differisce da All molto meno del 20% → LM ottiene risultati positivi molto sparsi, ma quando li ottiene è molto probabile che vi sia un campo.

T = Type
LM = Language Model
WT = Table Corpus (web table)

Esperimenti
PERFORMANCE F2FC



(a) WLISTS



(b) TDLISTS

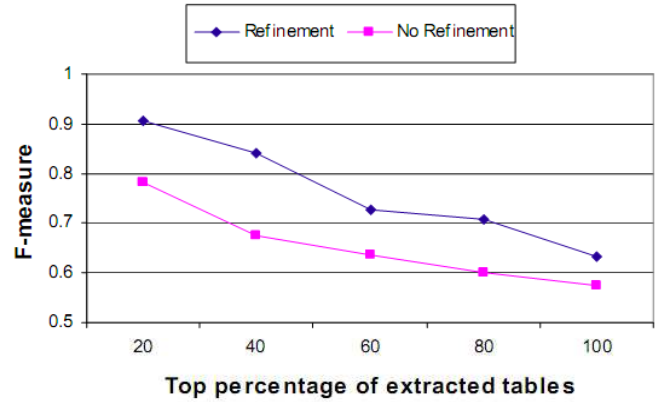
T C = Type Consistency
WC = Table Corpus Consistency
SC = Sintaxy Consistency
DC = Delimiter Consistency

In questo caso si considerano tutte le componenti di F2FQ.

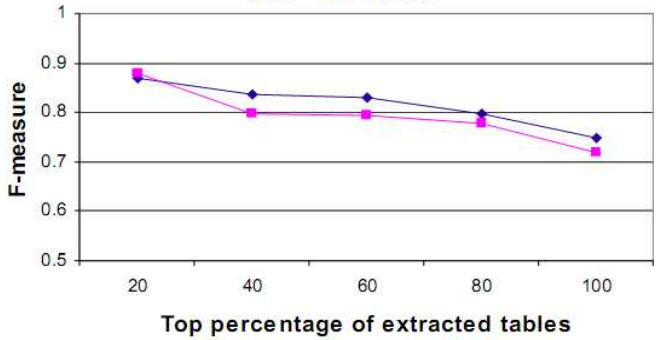
I grafici della f-measure mostrati considerano:
- tutte e tre le componenti assieme (All)
- le singole componenti separate (TC , WC, SC, DC)
- tutte le componenti escudendone di volta in volta una (All-TC, All-WC, All-SC, All-DC).

A volte considerare anche i delimitatori (All) porta ad un degrado dei risultati rispetto al non considerarli (All-DC), soprattutto nei TDLists perchè nelle liste di TDLists tutti i delimitatori sono spazi bianchi per costruzione e considerare altri possibili delimitatori come tali è un errore.

Esperimenti
PERFORMANCE RAFFINAMENTO



(a) WLists

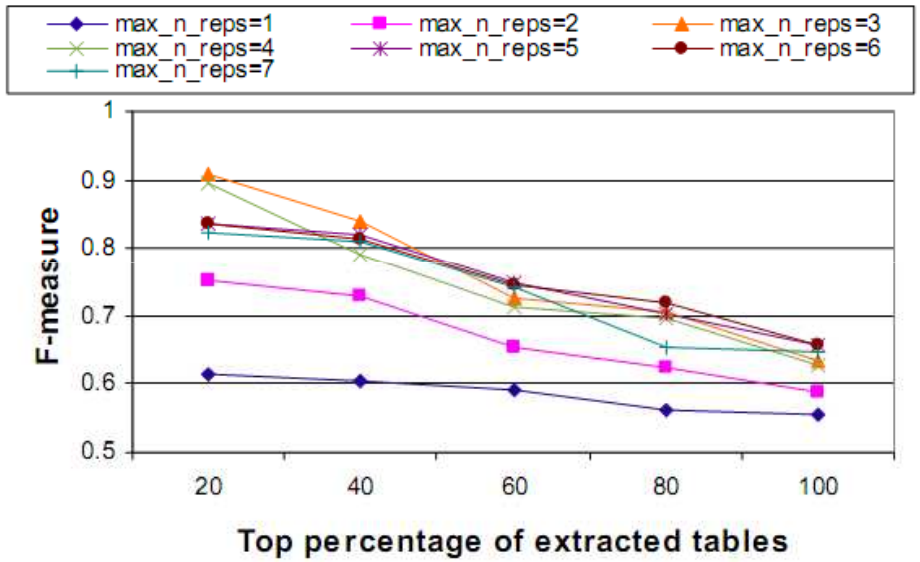


(b) TDLists

WLists
Il miglioramento portato dal raffinamento è migliore nel primo 20% e nel primo 40% delle tabelle estratte.
Se ci sono tanti campi splittati correttamente, ci sono pochi errori ed è quindi probabile che vengano corretti quasi tutti.

TDLists
Il miglioramento portato dal raffinamento non è mai superiore a 0,05 perchè le liste sono ottenute da delle tabelle quindi è più facile individuare lo split corretto già nella fase di split indipendente.

Esperimenti
PERFORMANCE FIELDS SUMMARIES



I valori di *max_n_reps* pari a 1 e 2 danno risultati peggiori;
dal valore 3 in poi aumentare non porta a significativi miglioramenti
→ per migliorare le prestazioni (calcolare meno F2FC) si sceglie un *max_n_reps* = 3.

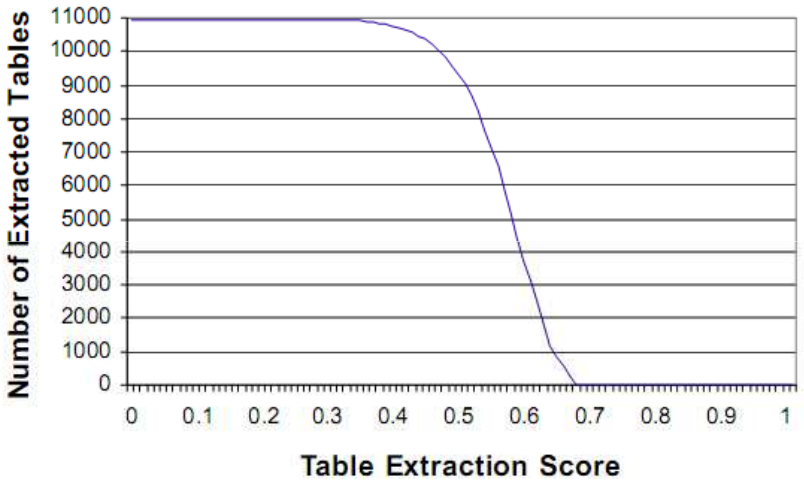
Esperimenti
CONSIDERAZIONI SU LARGA SCALA

Consideriamo un caso specifico: le liste contenute in 100.000 pagine web in inglese.

Queste vengono filtrate eliminando:
- liste con meno di 5 righe
- liste con più di 50 righe
- liste con più di 100 lettere per riga.

Rimangono 32.000 liste:
- 21.000 hanno solo una colonna (eliminate)
- 11.000 sono tabelle relazionali.

Di queste 11.000 si calcola il TE



1428 tabelle estratte hanno un TE superiore a 0,6 (buon risultato).
Dato che c'è in media una lista per pagina web, ci sono 1428 tabelle relazionali su 100000 liste → 1,4%.
Estendendo la stima a tutto il web (1mld di pagine) è probabile che ci siano 14mln di tabelle relazionali estraibili da liste nel web.

Sommario

Conclusioni

1. I risultati ottenuti suggeriscono che ci sia un grande numero di liste di qualità che possono essere estratte dal web.

2. Altri algoritmi che estraggono liste o altre informazioni dal web sono limitati da varie assunzioni che ne riducono il campo di applicabilità.

3. ListExtract non è sottoposto a questi limiti ed ha prestazioni migliori.

   In particolare ha come punti di forza di essere non supervisionato e indipendente dal dominio: questo fa sì che possa essere usato su larga scala nel web.