

An emerging paradigm in music distribution

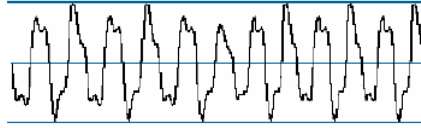
- End of an era for traditional distribution of music
 - MP3 + WWW = exchange of music in P2P
 - Is all this about piracy??
 - No! Internet as a new marketing and fulfillment model in the music industry
 - Consumers buying individual tracks
 - Growing industry: iTunes, iMusic
- MP3 + CBMIR + WWW = successful P2P nets
 - Only metadata/tags do not suffice
 - Queries based on humming (using a microphone) or on a small piece of musical file, are a more natural approach to MIR

Layout

- Background
- Problem definition
- Proposed method
- Experimental results
- Summary

Music data

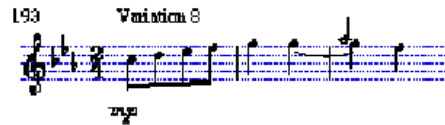
- Signal (wav, wma, mp3)



- Time-stamped events (midi)

```
t=0 NOn ch=1 num=72 vel=55 t=228 NOff ch=1 num=72
t=240 NOn ch=1 num=74 vel=57 t=468 NOff ch=1 num=74
t=480 NOn ch=1 num=75 vel=60 t=708 NOff ch=1 num=75
t=720 NOn ch=1 num=77 vel=64 t=948 NOff ch=1 num=77
t=960 NOn ch=1 num=79 vel=66 t=1416 NOff ch=1 num=79
t=1440 NOn ch=1 num=79 vel=70 t=2376 NOff ch=1 num=79
t=1920 NOn ch=1 num=80 vel=75 t=2832 NOff ch=1 num=80
t=2400 NOn ch=1 num=77 vel=73 t=2856 NOff ch=1 num=77
```

- Common Music Notation (score)



Similarity searching in audio data

- Ability to retrieve similar audio data by content
 - Content-Based Music Information Retrieval (CB-MIR)
- Query types
 - audio files
 - Query-By-Humming (QBH)
 - CMN



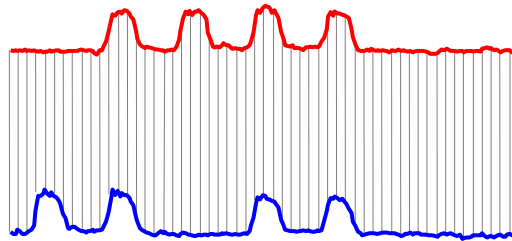
Similarity measures

- Similarity measure should allow for imprecise matches
- Similarity algorithm should be very efficient
- It should be possible to use the similarity algorithm efficiently in other computations, such as
 - Indexing
 - Subsequence similarity
 - clustering
 - rule discovery

Euclidean-like distances

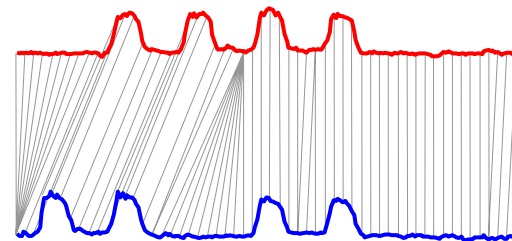
- Natural approach for time varying values (like audio)
- Simplistic
- Yet, more effective than many highly-specialized measures! (C. A. Ratanamahatana and E. Keogh. "Everything you know about Dynamic Time Warping is Wrong". 3rd Workshop on Mining Temporal and Sequential Data, 2004)

Euclidean Vs DTW



Euclidean Distance

Sequences are aligned "one to one".



"Warped" Time Axis

Nonlinear alignments are possible.

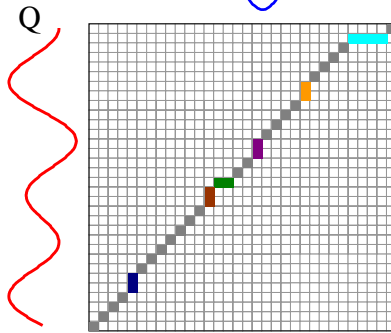
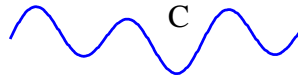
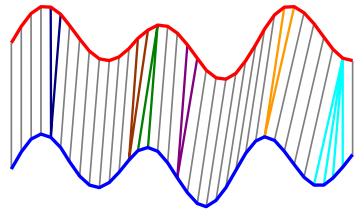
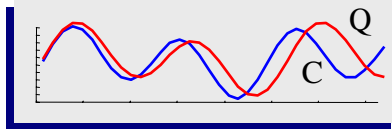
DTW

- Capability to withstand distortion of the comparing series in time axis.
- It allows for two locally out of phase pieces, that are nevertheless similar, to align in a non-linear manner.
- Different performances of same piece may include locally differentiated tempo (Large and Palmer, 2002)
- Has been used for MIR in centralized environments (Zhu and Shasha, 2003; Mazzoni and Danneberg, 2001; Jang et al., 2001; Adams et al., 2004)

How is DTW Calculated?

$$DTW(Q, C) = \min \left\{ \sqrt{\sum_{k=1}^K w_k} / K \right\}$$

$$\gamma(i, j) = d(q_i, c_j) + \min \{ \gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1) \}$$



Warping path w

Lower Bounding

We can speed up similarity search under DTW by using a lower bounding function.

Intuition

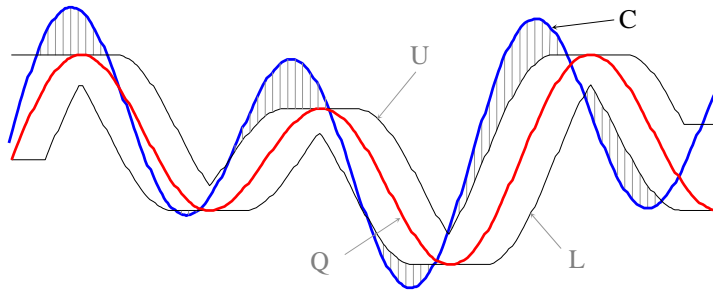
Try to use a cheap lower bounding calculation as often as possible.

Only do the expensive, full calculations when it is absolutely necessary.

Algorithm Lower_Bounding_Sequential_Scan(Q)

1. $best_so_far = \text{infinity};$
2. **for** all sequences in database
3. $LB_dist = \text{lower_bound_distance}(C_i, Q);$
4. **if** $LB_dist < best_so_far$
5. $true_dist = DTW(C_i, Q);$
6. **if** $true_dist < best_so_far$
7. $best_so_far = true_dist;$
8. $index_of_best_match = i;$
9. **endif**
10. **endif**
11. **endfor**

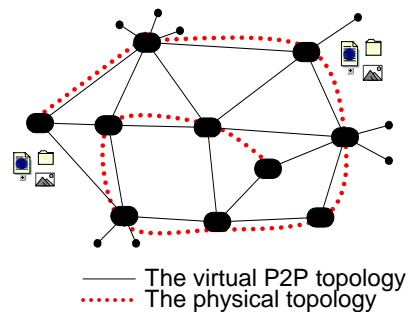
LB_Keogh



$$LB_Keogh(Q, C) = \sqrt{\sum_{i=1}^n \begin{cases} (c_i - U_i)^2 & \text{if } c_i > U_i & \text{(Case 1)} \\ (c_i - L_i)^2 & \text{if } c_i < L_i & \text{(Case 2)} \\ 0 & \text{otherwise} & \text{(Case 3)} \end{cases}}$$

Introduction to Peer-to-Peer

- Peer-to-Peer Computing definition:
"Sharing of computer resources and information through direct exchange"
- Clients (downloaders) are also servers
- Clients may join or leave the network at any time => highly **fault-tolerant** but with a cost!
- Searches are done within the virtual network while actual downloads are done offline (with HTTP).





Introduction to Peer-to-Peer

- Peer-to-Peer (P2P) systems are increasingly becoming popular.
- P2P file-sharing systems, such as Gnutella, Napster and Freenet realized a distributed infrastructure for sharing files.
- Traditionally, files were shared using the Client-Server model (e.g. http). Not scalable since they are centralized services.
- P2P uncover new advantages in simplicity of use, robustness, self organization and scalability.

Decentralized unstructured P2P networks

- Decentralized
 - Network failures due to central peer failure,
 - Impaired scalability,
 - joining/leaving of peers not easily handled,
 - Possible undesirable dominion of controllers
- Absence of structure
 - Looseness of control over data location,
 - Each peer shares its own documents without hosting any documents of other peers due to locality restrains

Related work

- The inaugural research paper dates back in 2002 (Wang et al., 2002)
- A hybrid configuration is presented in (Tzanetakis et al., 2004)
- (Yang, 2003) uses a feature selection and extraction process described in (Yang, 2002) for CBMIR in a decentralized unstructured P2P system.

Layout

- Background
- Problem definition
- Proposed method
- Experimental results
- Summary

Problem definition

- Searching, based on DTW, for similar acoustic data over unstructured decentralized P2P networks.
- Need for
 - Query representation
 - Query routing through P2P network
 - Computation of similarity at peers
 - False-alarm elimination at querer

Layout

- Background
- Problem definition
- Proposed method
- Experimental results
- Summary

Query representation (1)

- Propagate U, L (envelopes) of query Q
 - too large to travel around P2P!
- Sample to reduce traffic ($|Q| = N$, $|\text{sample}| = M$, $M \ll N$)
 - but, equi-depth sampling increases probability of entering case 3 in LB computation \Rightarrow many false alarms \Rightarrow increase traffic
- Alternative: focused sampling
 - Sample to increase probability of entering cases 1 or 2
 - Separate sampling of U and L
 - Sort U values, keep M smallest
 - Sort L values, keep M largest
- Why? Recall:
 - We enter case 1 when $C_i > U_i$. Smallest U_i values increase this probability
 - Analogously for case 2

Query representation (2)

- Query representation:
 - Focused samples of U and L
 - Plus, the indexes (offsets within sequences) from selected samples
- Total size: 4M
 - Overhead 2M over plain sample
- Represent selected positions with a bitmap of size N (1s at selected pos, 0s otherwise)
 - Better as long as $N < 32 M$ (see paper)
 - Suffices for samples larger than 3% (we use 10%)

Example of query representation

Seq. size: $N = 16$
 sample size: $M = 4$
 Window: $W = 4$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
U	235	9	73	93	224	75	96	71	81	209	131	19	121	213	57	55
L	135	5	56	83	141	25	6	15	21	109	121	14	111	112	37	5

BFSS-UNI	U'	0	1	2	3	L'	0	1	2	3	W = 4	Size
		235	224	81	121		135	141	21	111		$2M+1$

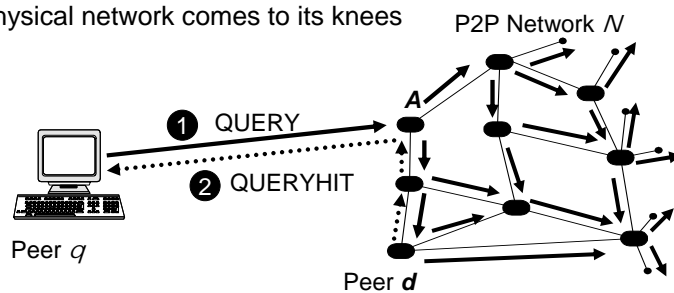
BFSS	U'	0	1	2	3	bitmap	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Size
		9	19	55	57			0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	
	L'	0	1	2	3	bitmap	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
		141	135	121	112			1	0	0	0	1	0	0	0	0	0	1	0	0	1	0	0

Query routing

Breadth-First Search (Gnutella)

- **Idea:** Each Query Message is propagated along all outgoing links of a peer using TTL (time-to-live).
- TTL is decremented on each forward until it becomes 0
- Technique for I.R in P2P systems such as Gnutella.
- **Highlights**

- The physical network comes to its knees



Similarity computation & False-alarm elimination

- Similarity is computed at each peer using the (sampled) LB.
- Querier receives all matches (with dist less than user-defined threshold), finds actual distances.

Selecting a sample size

- The main question is: how large samples to use?
- Large samples:
 - Less false-alarms => less traffic to return matches
 - Propagation of much larger U and L sequences => more traffic to propagate query representation
- Small samples:
 - Many false-alarms => more traffic to return matches
 - Propagation of much smaller U and L sequences => less traffic to propagate query representation
- Trade-off: examined experimentally

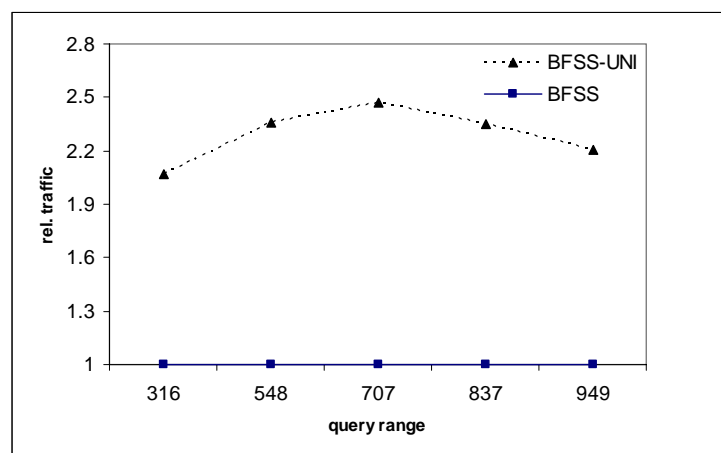
Layout

- Background
- Problem definition
- Proposed method
- Experimental results
- Summary

Experiments

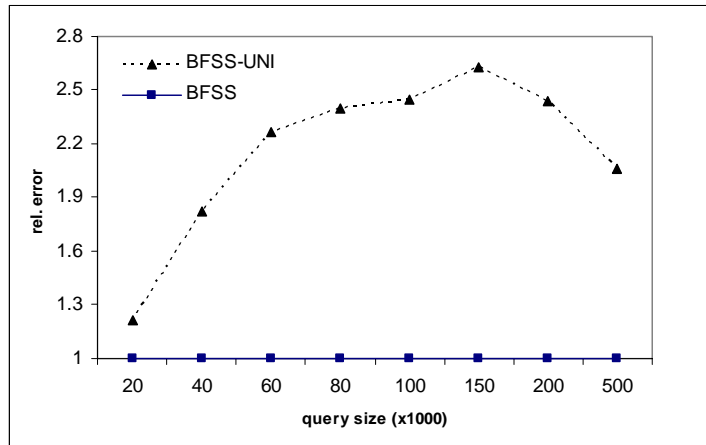
- Simulation test-bed:
 - 500 songs (various music genres, e.g., greek, pop, rock, classical) average length 5 min.
 - 100 network nodes
 - Average 7 neighboring nodes per node
 - Each sequence was randomly variably repeated with average value equal to 10.
- Compare:
 - Focused vs. uniform sampling
 - BFSS (proposed) vs. BFS (no sampling)
- Metrics:
 - average traffic (in MB)

BFSS vs. BFSS-UNI Relative traffic



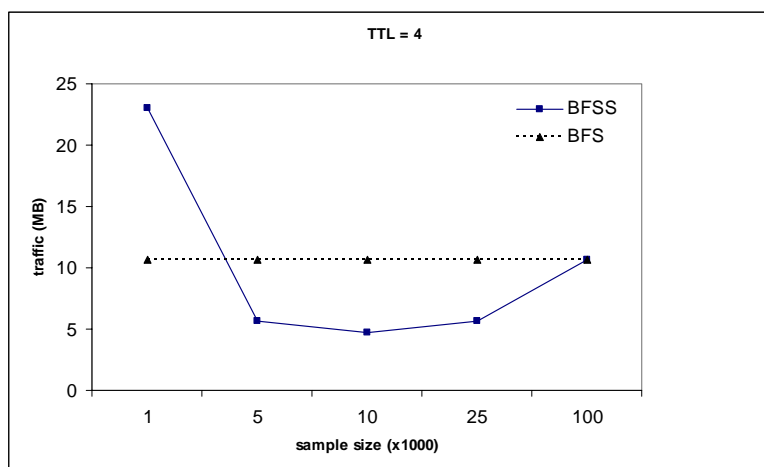
Uniform sampling produces bad underestimation of the LB

BFSS vs. BFSS-UNI Relative error w.r.t. actual LB Keogh value.



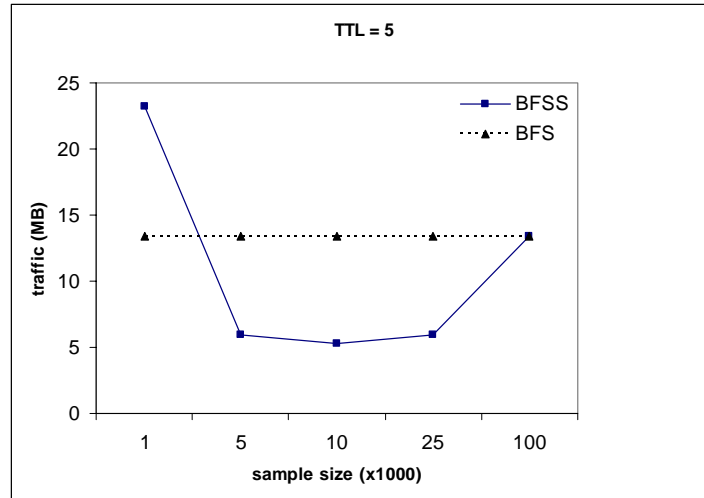
Uniform sampling produces bad underestimation of the LB

BFSS vs. BFS : Traffic while TTL=4

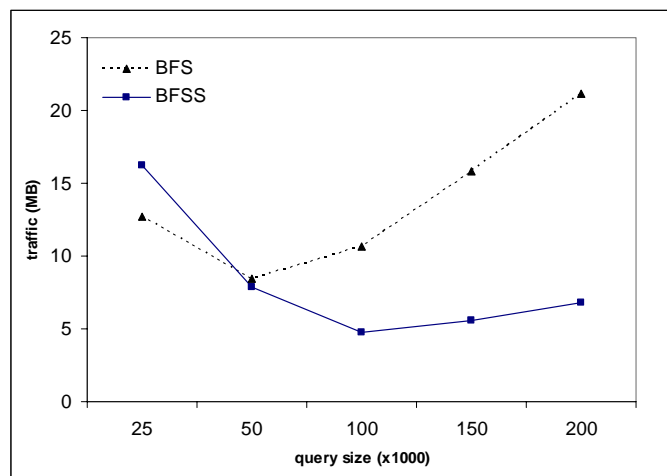


Small sample size => large number of false alarms => increased traffic

BFSS vs. BFS : Traffic while TTL=5

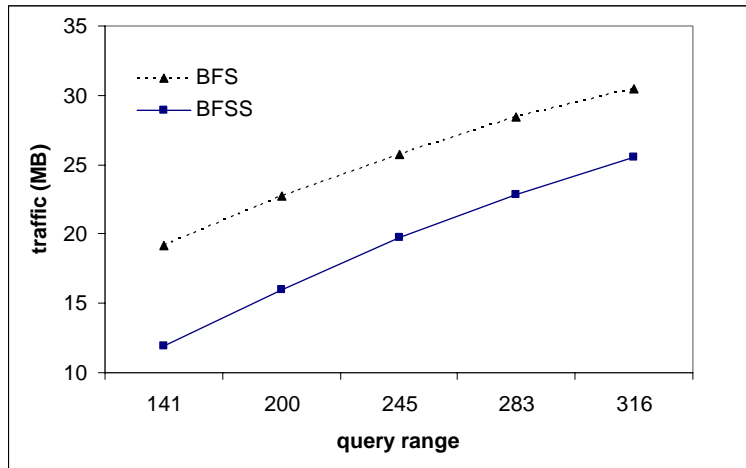


BFSS vs. BFS w.r.t. query size



Small queries => small sample size => many false alarms

BFSS vs. BFS w.r.t. query range.



Layout

- Background
- Problem definition
- Proposed method
- Experimental results
- Summary

Summary

- We have presented:
 - A novel algorithm, which efficiently retrieves similar audio data.
 - The proposed algorithm:
 - takes advantage of the absence of overhead in unstructured P2P networks
 - minimizes the required traffic using an intelligent sampling scheme
 - The design ensures no false negatives
 - Comparative evaluation showed significantly reduction in traffic

MUSICAL RETRIEVAL IN P2P NETWORKS UNDER THE WARPING DISTANCE

Thank you!

The algorithm

```
Procedure BFSS(Node  $n$ , int TTL, Sequence  $U'$ ,  
Sequence  $L'$ , float  $T_s$ )  
begin  
1. foreach data sequence  $D$  in  $n$   
2.   foreach phrase  $C$  of  $D$   
3.      $l = LB'(C, U', L')$   
4.     if  $l < T_s$   
5.       get query sequence  
6.       compute actual DTW distance,  $D$ ,  
         between phrase  $C$  and query sequence  
7.       if  $D \leq T_s$   
8.         include  $C$  in answer set  
9. if TTL > 0  
10.  foreach peer  $p$  of  $n$  that has not been visited yet  
11.   BFSS( $p$ , TTL-1,  $U'$ ,  $L'$ ,  $T_s$ )  
end
```