Artigos Clássicos em Computação Musical: Dannenberg e o acompanhamento musical em tempo real

Roberto Piassi Passos Bodo

Instituto de Matemática e Estatística
Universidade de São Paulo

13 de Março de 2013

O Artigo

"An On-Line Algorithm for Real-Time Accompaniment"

International Computer Music Conference de 1984

Roger B. Dannenberg

Carnegie Mellon University

Motivações

Motivações

- solucionar os problemas de sincronização inerentes ao acompanhamento por fita;

Motivações

- solucionar os problemas de sincronização inerentes ao acompanhamento por fita;
- aproveitar a flexibilidade oferecida pela síntese digital em tempo real;

Motivações

- solucionar os problemas de sincronização inerentes ao acompanhamento por fita;
- aproveitar a flexibilidade oferecida pela síntese digital em tempo real;
- surgimento de variadas aplicações musicais com diferentes níveis de complexidade;

Motivações

- solucionar os problemas de sincronização inerentes ao acompanhamento por fita;
- aproveitar a flexibilidade oferecida pela síntese digital em tempo real;
- surgimento de variadas aplicações musicais com diferentes níveis de complexidade;
- criar novas possibilidades artísticas ao considerar as sutilezas da performance do solista.

O Problema

O Problema

Criar um sistema tal que o computador tenha a habilidade de acompanhar um músico solista.

O Problema

Criar um sistema tal que o computador tenha a habilidade de acompanhar um músico solista.

Dannenberg o dividiu em 3 sub-problemas:

O Problema

Criar um sistema tal que o computador tenha a habilidade de acompanhar um músico solista.

Dannenberg o dividiu em 3 sub-problemas:

1. processar a entrada do solista em tempo real;

O Problema

Criar um sistema tal que o computador tenha a habilidade de acompanhar um músico solista.

Dannenberg o dividiu em 3 sub-problemas:

1. processar a entrada do solista em tempo real;
2. comparar essa entrada com a partitura original;

O Problema

Criar um sistema tal que o computador tenha a habilidade de acompanhar um músico solista.

Dannenberg o dividiu em 3 sub-problemas:

1. processar a entrada do solista em tempo real;
2. comparar essa entrada com a partitura original;
3. gerar o acompanhamento de acordo com o andamento do músico.

Foco do artigo

Foco do artigo

No segundo sub-problema, ou seja, casamento entre os eventos da performance e os da partitura.

Foco do artigo

No segundo sub-problema, ou seja, casamento entre os eventos da performance e os da partitura.

Problemas relacionados (fora do domínio da música):

Foco do artigo

No segundo sub-problema, ou seja, casamento entre os eventos da performance e os da partitura.

Problemas relacionados (fora do domínio da música):

Encontrar diferenças entre dois arquivos de computador/encontrar a melhor correspondência entre sequências de símbolos.

Foco do artigo

No segundo sub-problema, ou seja, casamento entre os eventos da performance e os da partitura.

Problemas relacionados (fora do domínio da música):

Encontrar diferenças entre dois arquivos de computador/encontrar a melhor correspondência entre sequências de símbolos.

- UNIX diff;

Foco do artigo

No segundo sub-problema, ou seja, casamento entre os eventos da performance e os da partitura.

Problemas relacionados (fora do domínio da música):

Encontrar diferenças entre dois arquivos de computador/encontrar a melhor correspondência entre sequências de símbolos.

- UNIX diff;
  Tempo proporcional ao comprimento da partitura para cada match.

Foco do artigo

No segundo sub-problema, ou seja, casamento entre os eventos da performance e os da partitura.

Problemas relacionados (fora do domínio da música):

Encontrar diferenças entre dois arquivos de computador/encontrar a melhor correspondência entre sequências de símbolos.

- UNIX diff;
  Tempo proporcional ao comprimento da partitura para cada match.
- Papers sobre reconhecimento de fala.

Foco do artigo

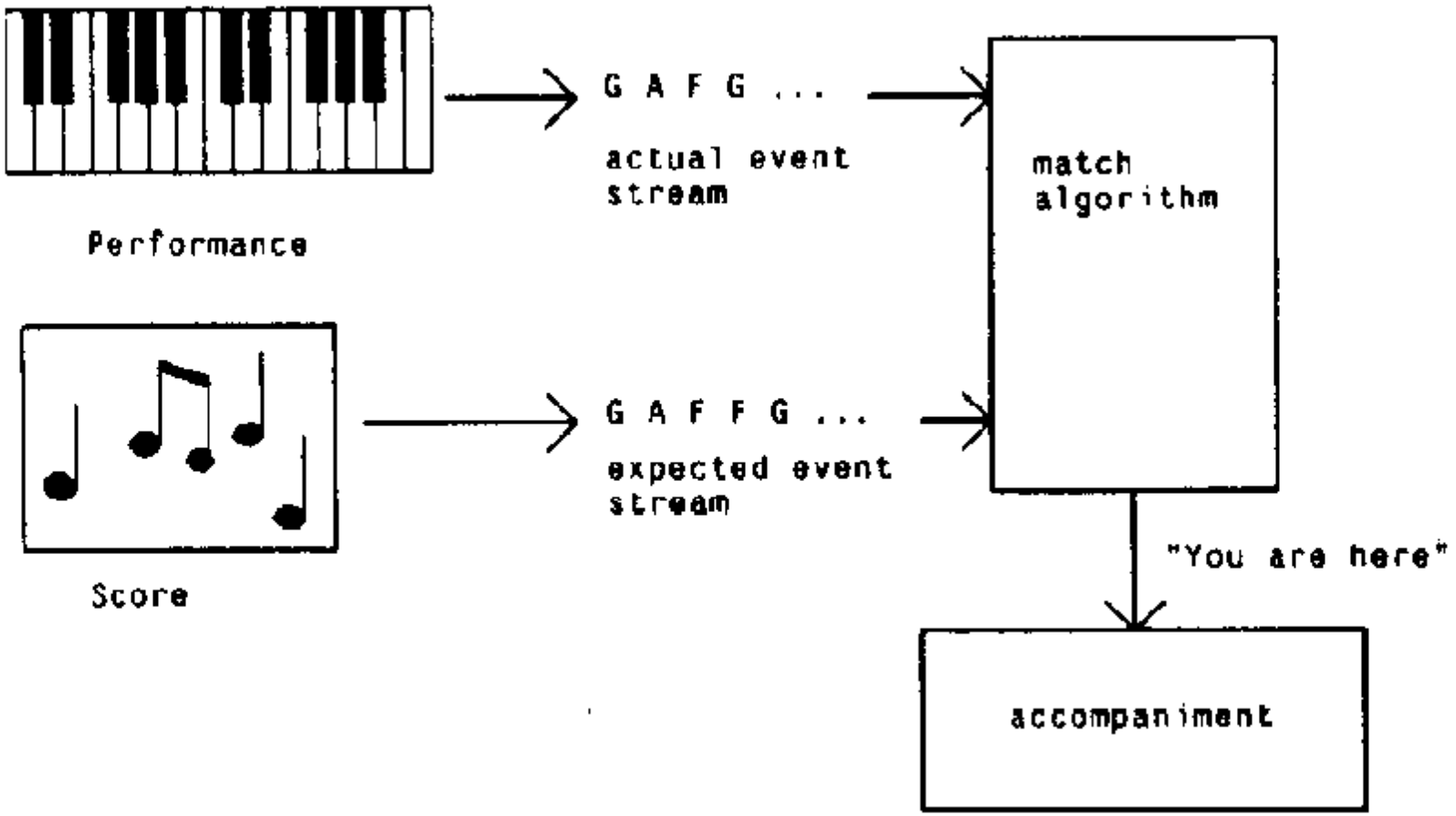
No segundo sub-problema, ou seja, casamento entre os eventos da performance e os da partitura.

Problemas relacionados (fora do domínio da música):

Encontrar diferenças entre dois arquivos de computador/encontrar a melhor correspondência entre sequências de símbolos.

- UNIX diff;
  Tempo proporcional ao comprimento da partitura para cada match.
- Papers sobre reconhecimento de fala.
  Técnica de programação dinâmica utilizada no casamento de features obtidas de gravações de falas com templates pré-conhecidos.

O modelo


Performance
G A F G ...
actual event stream
match algorithm
Score
G A F F G ...
expected event stream
"You are here"
accompaniment

1. Entrada

1. Entrada

Considerações:

1. Entrada

Considerações:

- conseguimos reconhecer um número limitado de eventos;

1. Entrada

Considerações:

- conseguimos reconhecer um número limitado de eventos;
- é simples determinarmos se eventos são iguais ou diferentes;

1. Entrada

Considerações:

- conseguimos reconhecer um número limitado de eventos;
- é simples determinarmos se eventos são iguais ou diferentes;
- a entrada do solista é um stream de eventos;

1. Entrada

Considerações:

- conseguimos reconhecer um número limitado de eventos;
- é simples determinarmos se eventos são iguais ou diferentes;
- a entrada do solista é um stream de eventos;
- a partitura é uma lista ordenada dos eventos esperados;

2. Algoritmo de matching

2. Algoritmo de matching

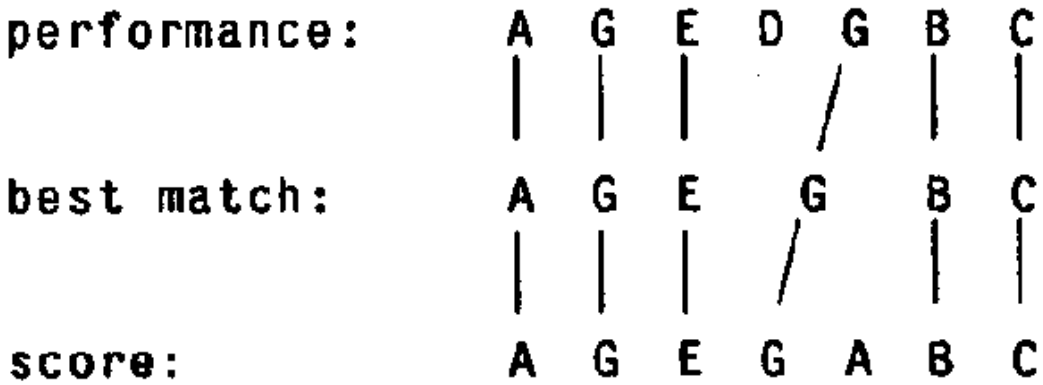
Intenção: encontrar o *melhor casamento* entre a entrada e a partitura do solista

2. Algoritmo de matching

Intenção: encontrar o *melhor casamento* entre a entrada e a partitura do solista, onde *melhor casamento* = maior subsequência comum.

2. Algoritmo de matching

Intenção: encontrar o *melhor casamento* entre a entrada e a partitura do solista, onde *melhor casamento* = maior subsequência comum.



Programação Dinâmica

= "recursão com tabela"

Programação Dinâmica

= "recursão com tabela"

= "divisão-e-conquista sequencial"

Programação Dinâmica

= "recursão com tabela"

= "divisão-e-conquista sequencial"

= "transformação inteligente de recursão em iteração"

Números de Fibonacci

```
FIBONACCI-REC(n)
1. if n <= 1
2.         then return n
3.         else a = FIBONACCI-REC(n - 1)
4.                 b = FIBONACCI-REC(n - 2)
5.                 return a + b
```

Números de Fibonacci

```
FIBONACCI-REC(n)
1. if n <= 1
2.         then return n
3.         else a = FIBONACCI-REC(n - 1)
4.                 b = FIBONACCI-REC(n - 2)
5.                 return a + b
```

```
FIBONACCI(n)
1. f[0] = 0
2. f[1] = 1
3. for i = 2 to n do
4.         f[i] = f[i - 1] + f[i - 2]
5. return f[n]
```

Programação dinâmica

Programação dinâmica

do CLRS

Programação dinâmica

do CLRS

- caracterizar a estrutura da solução ótima;

Programação dinâmica

do CLRS

- caracterizar a estrutura da solução ótima;
- definir recursivamente o valor de uma solução ótima;

Programação dinâmica

do CLRS

- caracterizar a estrutura da solução ótima;
- definir recursivamente o valor de uma solução ótima;
- calcular o valor de uma solução ótima de maneira bottom-up;

Programação dinâmica

do CLRS

- caracterizar a estrutura da solução ótima;
- definir recursivamente o valor de uma solução ótima;
- calcular o valor de uma solução ótima de maneira bottom-up;
- construir a solução ótima a partir dos resultados obtidos.

Maior subsequência comum

dos slides do IME

$X = A\ B\ C\ B\ D\ A\ B$

$Y = B\ D\ C\ A\ B\ A$

Maior subsequência comum

dos slides do IME

X = A B C B D A B

Y = B D C A B A

uma subsequência comum: B D A

Maior subsequência comum

dos slides do IME

X = A B C B D A B

Y = B D C A B A

uma subsequência comum: B D A

outra subsequência comum: B C B

Maior subsequência comum

dos slides do IME

X = A B C B D A B

Y = B D C A B A

uma subsequência comum: B D A

outra subsequência comum: B C B

uma subsequência comum máxima: B C A B

Maior subsequência comum

dos slides do IME

X = A B C B D A B

Y = B D C A B A

uma subsequência comum: B D A

outra subsequência comum: B C B

uma subsequência comum máxima: B C A B

outra subsequência comum máxima: B D A B

Maior subsequência comum

do CLRS

Teorema 6: Seja X = x[1], x[2], ..., x[m] e Y = y[1], y[2], ..., y[n] duas sequências e seja Z = z[1], z[2], ..., z[k] uma maior subsequência comum a X e Y.

- Se x[m] = y[n], então z[k] = x[m] = y[n] e Z(k-1) é uma maior subseqência comum a X(m-1) e Y(n-1);
- Se x[m] = y[n] e z[k] = x[m], então Z é uma maior subsequência comum a X(m-1) e Y;
- Se x[m] = y[n] e z[k] = y[n] , então Z é uma maior subsequência comum a X e Y(n-1).

Maior subsequência comum

do CLRS

Solução recursiva:

- se x[m] = y[n], devemos achar a maior subsequência comum a X(m-1) e Y(n-1); concatenando x[m] = y[n] a essa subsequência temos a maior subsequência comum a X e Y.
- se x[m] = y[n], devemos achar a maior subsequência comum a X(m-1) e Y, e a X e Y(n-1). A maior delas será a maior subsequência comum a X e Y.

Maior subsequência comum

```
LCS_LENGTH(X, m, Y, n)
for i = 1 to m do
        for j = 1 to n do
                if x[i - 1] == y[j - 1]
                        then c[i][j] = c [i - 1][j - 1] + 1
                        else if c[i - 1][j] >= c[i][j - 1]
                                then c[i][j] = c[i - 1][j]
                                else c[i][j] = c[i][j - 1]
```

Adaptações

- cada linha corresponde a um evento da partitura;

Adaptações

- cada linha corresponde a um evento da partitura;
- cada coluna corresponde a um evento do solista;

Adaptações

- cada linha corresponde a um evento da partitura;
- cada coluna corresponde a um evento do solista;
- a cada evento detectado um nova coluna é calculada.

Adaptações

- cada linha corresponde a um evento da partitura;
- cada coluna corresponde a um evento do solista;
- a cada evento detectado um nova coluna é calculada.

```
          performance:

             A G E D G B C
score:    A  1 1 1 1
          G  1 2 2 2
          E  1 2 3 3
          G  1 2 3 3
          A  1 2 3 3
          B  1 2 3 3
          C  1 2 3 3
```

Adaptações

Valor da i-ésima linha e j-ésima coluna: o tamanho da maior subsequência comum.

Adaptações

Valor da i-ésima linha e j-ésima coluna: o tamanho da maior subsequência comum.

Como saber onde estamos na partitura para tocarmos o acompanhamento?

Adaptações

Valor da i-ésima linha e j-ésima coluna: o tamanho da maior subsequência comum.

Como saber onde estamos na partitura para tocarmos o acompanhamento?

A cada atualização do valor máximo, um novo sincronismo da performance com a partitura.

Adaptações

Valor da i-ésima linha e j-ésima coluna: o tamanho da maior subsequência comum.

Como saber onde estamos na partitura para tocarmos o acompanhamento?

A cada atualização do valor máximo, um novo sincronismo da performance com a partitura.

	performance:						
		A	G	E	D	G	B	C
score:	A	**1**	1	1	1	1	1	1
	G	1	**2**	2	2	2	2	2
	E	1	2	**3**	3	3	3	3
	G	1	2	3	3	**4**	4	4
	A	1	2	3	3	4	4	4
	B	1	2	3	3	4	**5**	5
	C	1	2	3	3	4	5	**6**

Adaptações

Economizando memória:

Adaptações

Economizando memória:

- armazenar apenas as colunas anterior e atual.

Adaptações

Economizando memória:

- armazenar apenas as colunas anterior e atual.

Economizando tempo:

Adaptações

Economizando memória:

- armazenar apenas as colunas anterior e atual.

Economizando tempo:

- utilizar apenas uma pequena janela em torno da nota esperada.

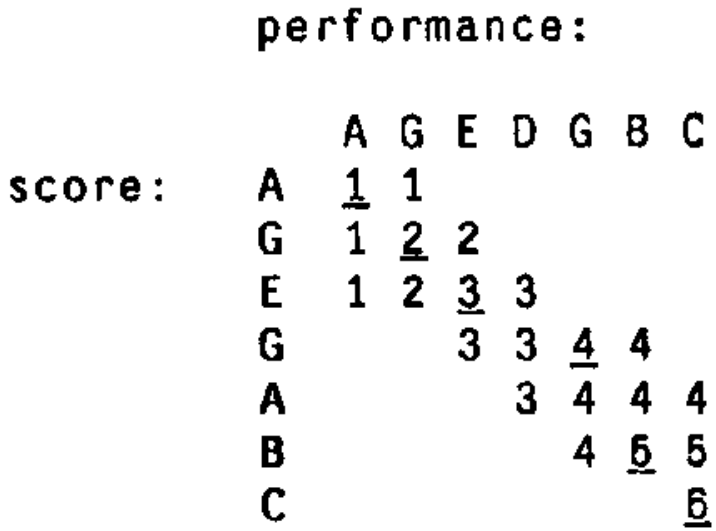
Adaptações

Economizando memória:

- armazenar apenas as colunas anterior e atual.

Economizando tempo:

- utilizar apenas uma pequena janela em torno da nota esperada.



Heurísticas

Heurísticas

(1) O centro da janela é determinado da seguinte forma:

Heurísticas

(1) O centro da janela é determinado da seguinte forma:

- se temos um match, centralizar a janela na linha que o match ocorreu;

Heurísticas

(1) O centro da janela é determinado da seguinte forma:

- se temos um match, centralizar a janela na linha que o match ocorreu;
- caso contrário, descer a janela uma coluna.

Heurísticas

(1) O centro da janela é determinado da seguinte forma:

- se temos um match, centralizar a janela na linha que o match ocorreu;
- caso contrário, descer a janela uma coluna.

performance:

score:

	A	G	E	D	G	B	C
A	1	1					
G	1	2	2				
E	1	2	3	3			
G			3	3	4	4	
A				3	4	4	4
B					4	5	5
C							6

Heurísticas

(1) O centro da janela é determinado da seguinte forma:

- se temos um match, centralizar a janela na linha que o match ocorreu;
- caso contrário, descer a janela uma coluna.

```
                performance:

                   A  G  E  D  G  B  C
score:          A  1  1
                G  1  2  2
                E  1  2  3  3
                G        3  3  4  4
                A           3  4  4  4
                B              4  5  5
                C                    6
```

Resultado obtido: a janela acompanha a partitura até o final se o número de eventos detectados for igual ao número de eventos esperado.

Heurísticas

(2) O tamanho da janela deverá ser maior ou igual a (2 * n) + 1, onde n é o número de erros consecutivos tolerados.

Heurísticas

(2) O tamanho da janela deverá ser maior ou igual a (2 * n) + 1, onde n é o número de erros consecutivos tolerados.

performance:

score:		A	G	E	D	G	B	C
	A	1	1					
	G	1	2	2				
	E	1	2	3	3			
	G			3	3	4	4	
	A				3	4	4	4
	B					4	5	5
	C							6

Heurísticas

(2) O tamanho da janela deverá ser maior ou igual a (2 * n) + 1, onde n é o número de erros consecutivos tolerados.

```
          performance:

             A  G  E  D  G  B  C
score:    A  1  1
          G  1  2  2
          E  1  2  3  3
          G        3  3  4  4
          A           3  4  4  4
          B              4  5  5
          C                    6
```

Resultado obtido: a partitura evolui mesmo com uma sequência de notas erradas.

Heurísticas

(3) O centro da janela só é permitido avançar duas linhas no máximo por evento detectado.

Heurísticas

(3) O centro da janela só é permitido avançar duas linhas no máximo por evento detectado.

F B G
B
G
E
F
C
D

Heurísticas

(3) O centro da janela só é permitido avançar duas linhas no máximo por evento detectado.

Resultado obtido: evitamos pulos muito grandes na partitura.

Heurísticas

(4) Associar uma penalidade a um match que pula eventos da partitura.

Heurísticas

(4) Associar uma penalidade a um match que pula eventos da partitura.

	F	B	G
B			
G			
E			
F			
C			
B			
D			
G			

Heurísticas

(4) Associar uma penalidade a um match que pula eventos da partitura.

	F	B	G
B			
G			
E			
F			
C			
B			
D			
G			

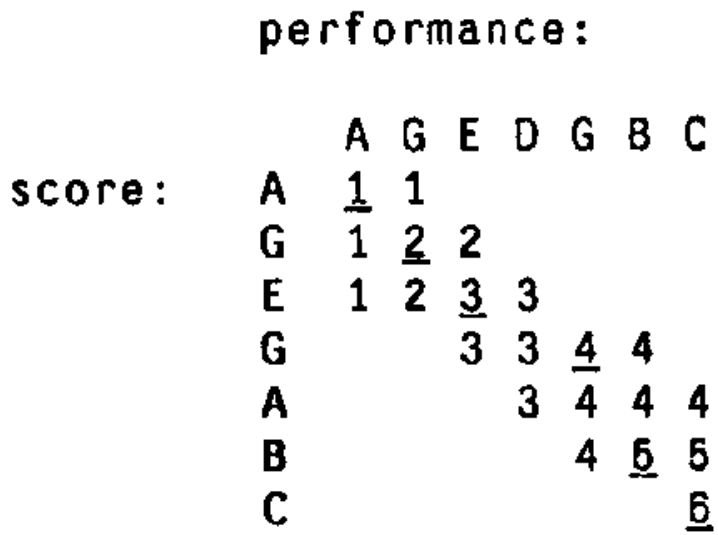
Resultado obtido: evitamos pulos longos consecutivos.

Heurísticas

(5) Sempre considerar o casamento como sendo o evento mais cedo da partitura.

Heurísticas

(5) Sempre considerar o casamento como sendo o evento mais cedo da partitura.



Heurísticas

(5) Sempre considerar o casamento como sendo o evento mais cedo da partitura.

performance:

score:		A	G	E	D	G	B	C
	A	1	1					
	G	1	2	2				
	E	1	2	3	3			
	G			3	3	4	4	
	A				3	4	4	4
	B					4	5	5
	C							6

Resultado obtido: resolvemos a dúvida quando temos o valor máximo aparecendo em várias linhas de uma mesma coluna.

3. Acompanhamento

3. Acompanhamento

O algoritmo de matching irá nos informar a localização temporal da performance em relação a partitura.

3. Acompanhamento

O algoritmo de matching irá nos informar a localização temporal da performance em relação a partitura.

Iremos utilizar essa informação para gerar o acompanhamento (lista ordenada de eventos).

3. Acompanhamento

O algoritmo de matching irá nos informar a localização temporal da performance em relação a partitura.

Iremos utilizar essa informação para gerar o acompanhamento (lista ordenada de eventos).

Exemplo: *mid2csv*
2, 79, Note_on_c, 0, 64, 127
2, 139, Note_on_c, 0, 65, 111
2, 379, Note_on_c, 0, 72, 126
2, 439, Note_on_c, 0, 69, 110

Relógio Virtual

Relógio Virtual

Exemplo: um player de MIDI

Relógio Virtual

Exemplo: um player de MIDI

Um relógio será inicializado e os eventos serão tocados cronologicamente.

Relógio Virtual

Exemplo: um player de MIDI

Um relógio será inicializado e os eventos serão tocados cronologicamente.

Suponha que podemos variar a velocidade desse relógio. Assim poderemos tocar o acompanhamento mais rápido ou mais devagar.

Relógio Virtual

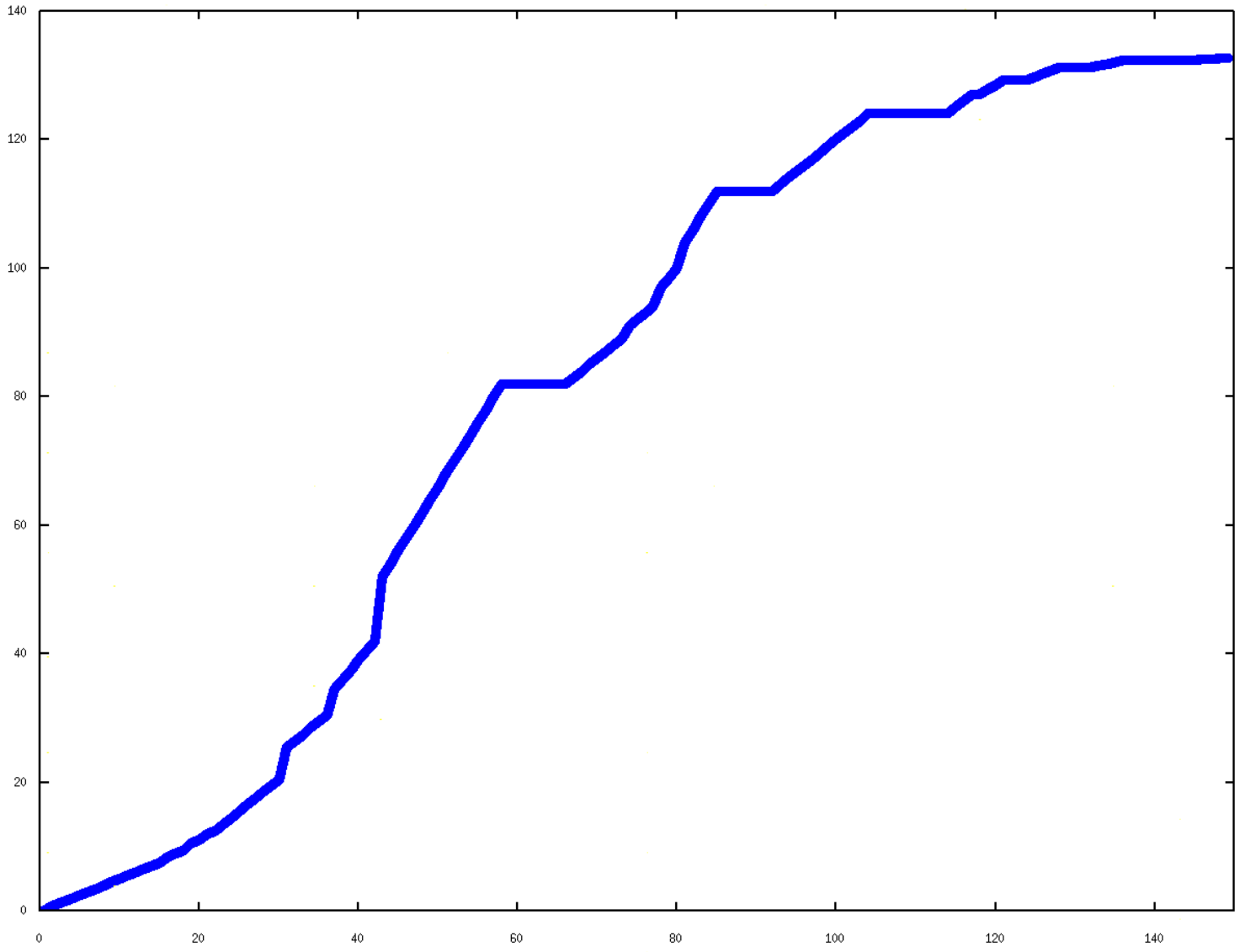
Exemplo: um player de MIDI

Um relógio será inicializado e os eventos serão tocados cronologicamente.

Suponha que podemos variar a velocidade desse relógio. Assim poderemos tocar o acompanhamento mais rápido ou mais devagar.

V = (R - Rref)*S + Vref

Tempo Real vs. Tempo Virtual



Limitações

Limitações

- ajustes de tempo são graduais e não seguem nenhuma regra musical;

Limitações

- ajustes de tempo são graduais e não seguem nenhuma regra musical;
- a entrada é uma lista ordenada de eventos e, assim, o sistema não sabe lidar com:

Limitações

- ajustes de tempo são graduais e não seguem nenhuma regra musical;
- a entrada é uma lista ordenada de eventos e, assim, o sistema não sabe lidar com:
  - instrumentos polifônicos;

Limitações

- ajustes de tempo são graduais e não seguem nenhuma regra musical;
- a entrada é uma lista ordenada de eventos e, assim, o sistema não sabe lidar com:
  - instrumentos polifônicos;
  - improvisação;

Limitações

- ajustes de tempo são graduais e não seguem nenhuma regra musical;
- a entrada é uma lista ordenada de eventos e, assim, o sistema não sabe lidar com:
  - instrumentos polifônicos;
  - improvisação;
  - trinados;

Limitações

- ajustes de tempo são graduais e não seguem nenhuma regra musical;
- a entrada é uma lista ordenada de eventos e, assim, o sistema não sabe lidar com:
  - instrumentos polifônicos;
  - improvisação;
  - trinados;
  - glissandos.

That's all folks!

Obrigado! =)