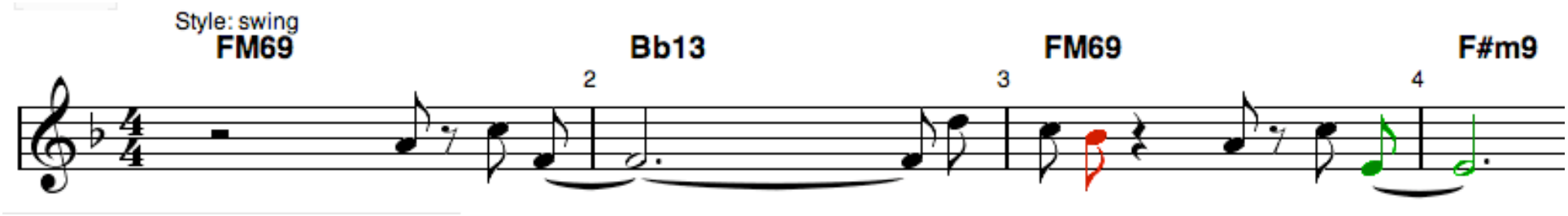
Blues for Gary



Mini-Languages for Impro-Visor

Robert Keller,

David Morrison, Stephen Jones,
Martin Hunt, Aaron Wolin

Harvey Mudd College

Steve Gomez

Dartmouth College

Support Acknowledgment

- Mellon Foundation
  Faculty Enhancement Grant

- NSF REU
  (Research Experience for Undergraduates)

Domains

- Human-Computer Interfaces
- Data Language Design
- Jazz Improvisation
- Music Education
- Melody Generation AI

Jazz Improvisation

- Scenario: A jazz group playing a standard tune.
- "**Head**": The musicians play an arrangement of the original melody.
- "**Choruses**": Some or all musicians improvise solos over the same, or related, chord changes to the original
- Throughout: Rhythm section improvises accompaniment.

Improvisation Techniques

- Most musicians employ, to varying extents, a *practiced* vocabulary of melodic fragments called “licks”.
- Most study solos of other musicians to get ideas.
- Sometimes these solos are **transcribed** or **memorized**, but rarely performed intact.
- Some pre-construct their own solos, to be treated as above.

Impro-Visor (Improvisation Advisor)

- Software to enhance the user's ability to
  - Improvise interesting solos.
  - Understand tunes and jazz harmony.
- By
  - *Engaging* him/her in the act of creation.
  - Providing support in the form of musical knowledge ("advice").
  - Generating melodies as usable examples.

Impro-Visor Usage

- User supplies chord sequence.
- Impro-Visor supplies musical knowledge, such as licks.
- User can save licks for posterity.
- Can also be used for transcription, composition.

Related Work: “Band-in-a-Box”

- Commercial software oriented to practice or, marginally, performance accompaniment:
  - Able generate entire “improvised” solo without user’s involvement (not our main purpose).
  - A proprietary rote **database** seems to be used.
  - Elements of the solo recur after awhile.

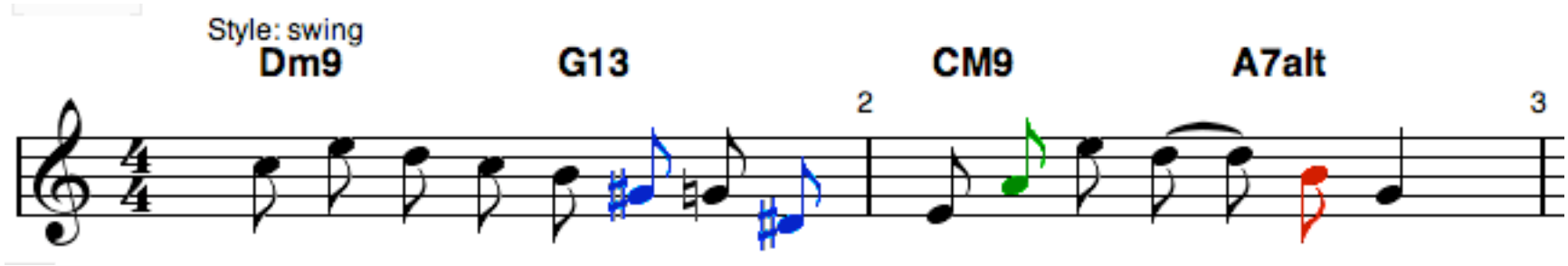
Related Work: “Band-out-of-a-Box”

- Belinda Thom’s PhD Thesis
  - Tried to use statistical theory to learn style of soloist.
  - Create companion for “trading fours”.

Related Work: “GenJam”

- Created by Al Biles, RIT.
- Generate licks by genetic algorithm.
- React to soloist in real-time.
- One man + one computer band.

Example Solo Fragment in Impro-Visor



Optional **color coding** for visual feedback:

black: chord tone

green: “color” tone (aka “tension”)

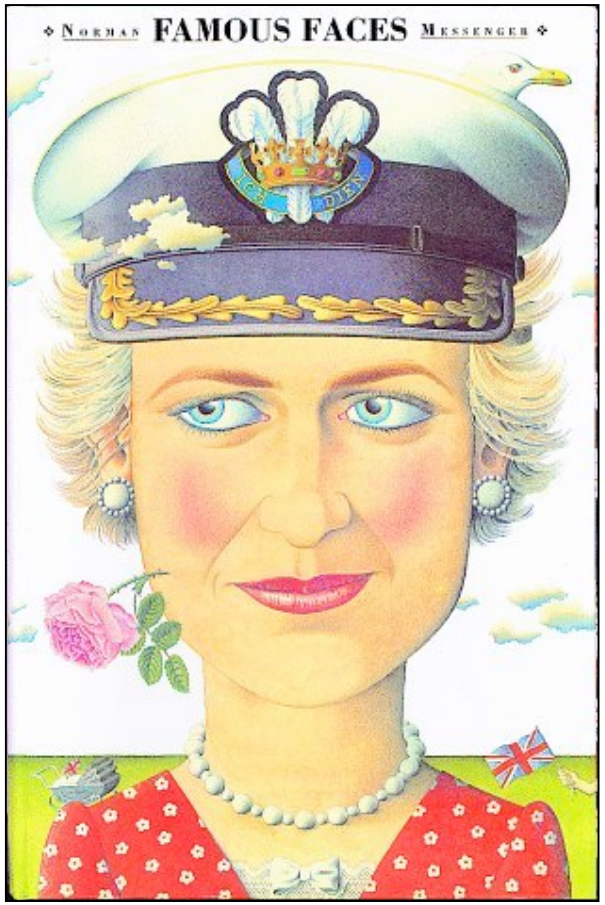
blue: approach tone to one of the above

red: none of the above (error?)

Principle of Modularity

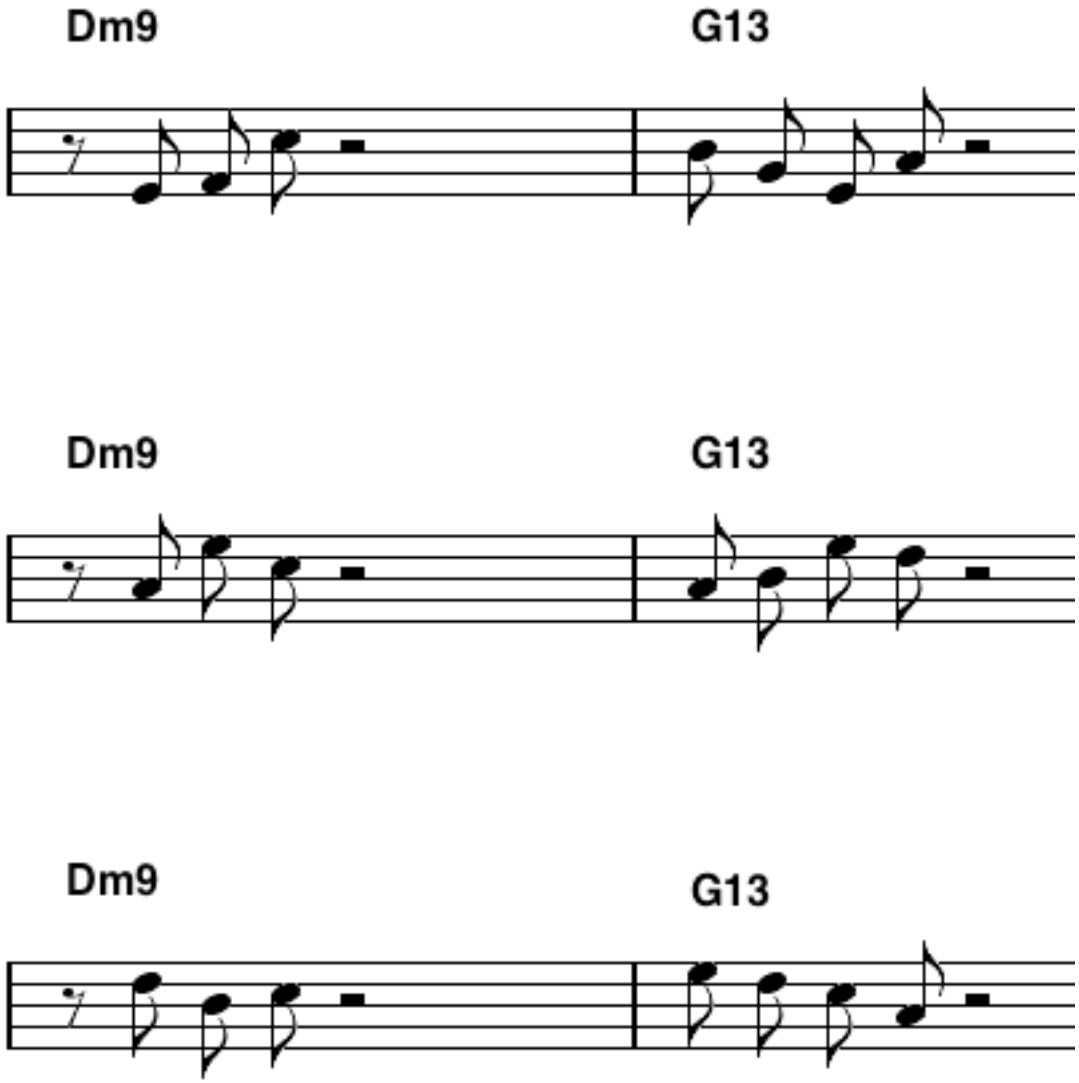
- Solos can be constructed by concatenating library segments:
  - **Cells** (over one chord, uniform note durations)
  - **Idioms** (over one chord, familiar, non-uniform)
  - **Licks** (over one or more chords)
  - **Quotes** (borrowed from another song or solo)
- Like “mix-and-match” book or Mozart’s dice game

Mix and Match Book

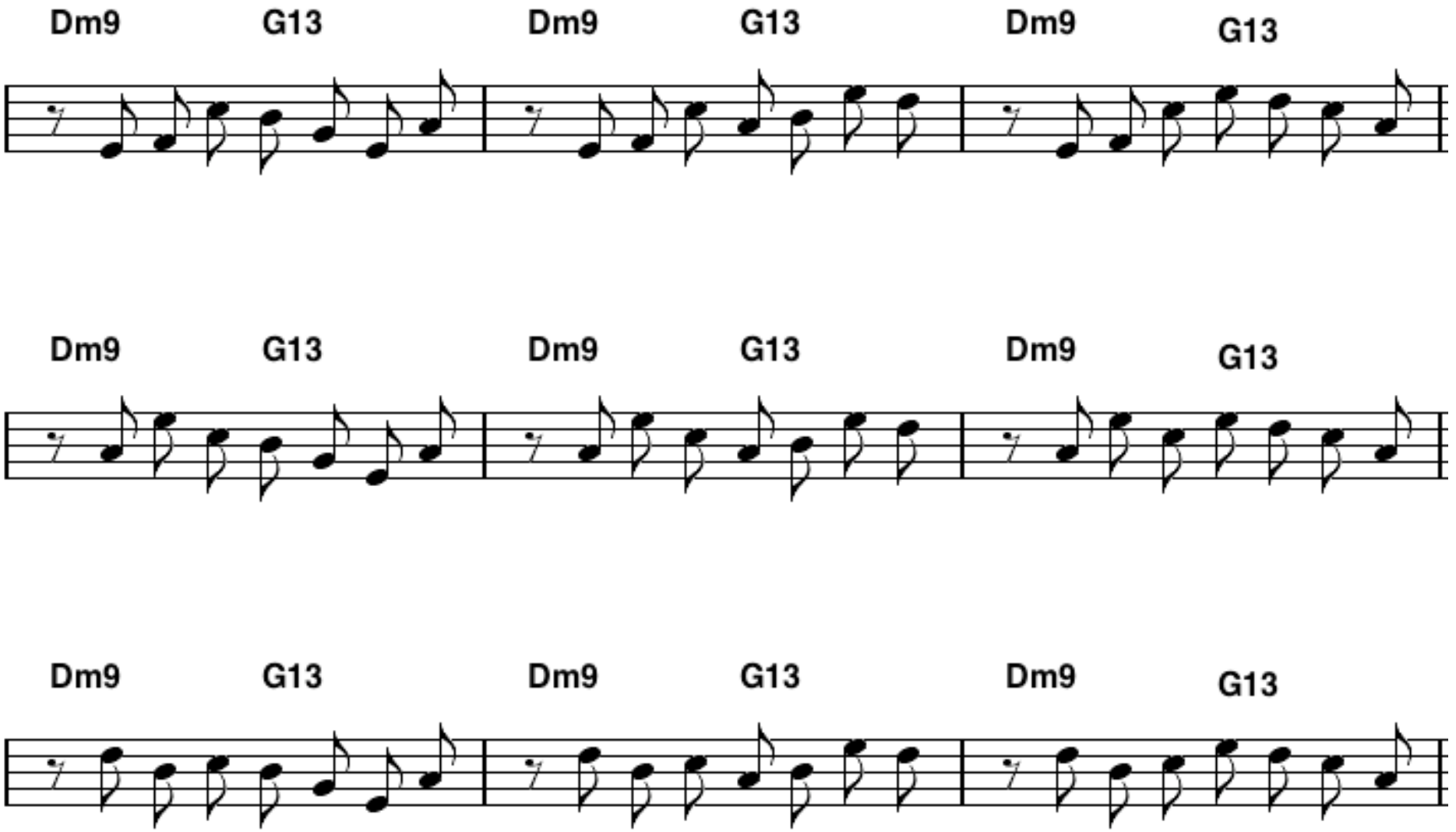


Six Cells over two chords …

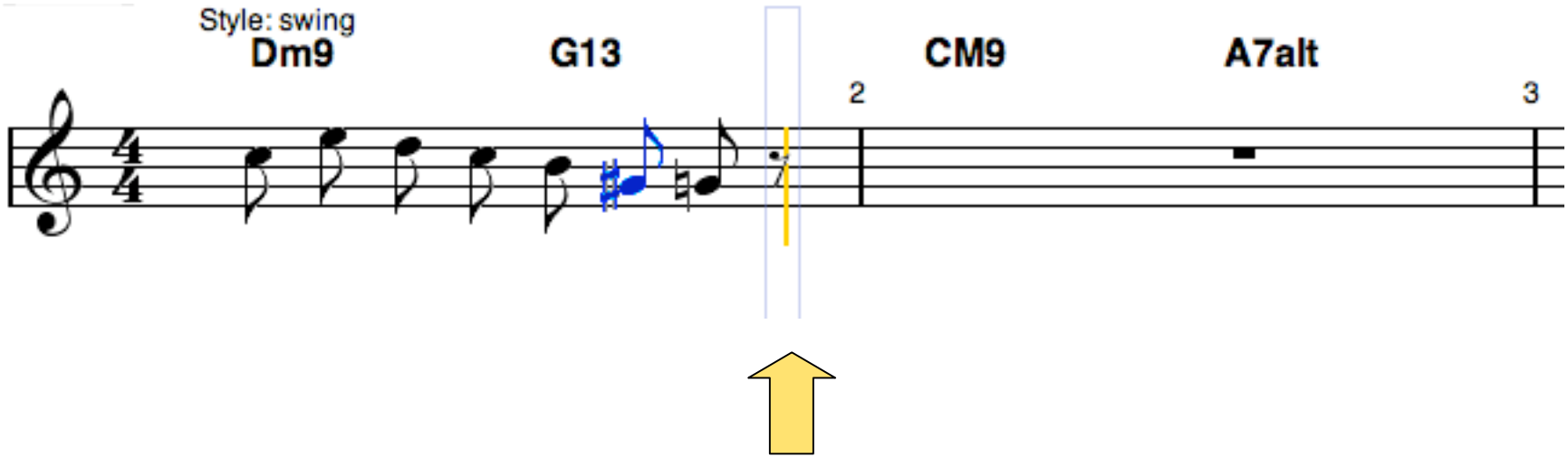
(Dm9 G13 are examples of two chords often found in succession.)



… produce nine different licks.

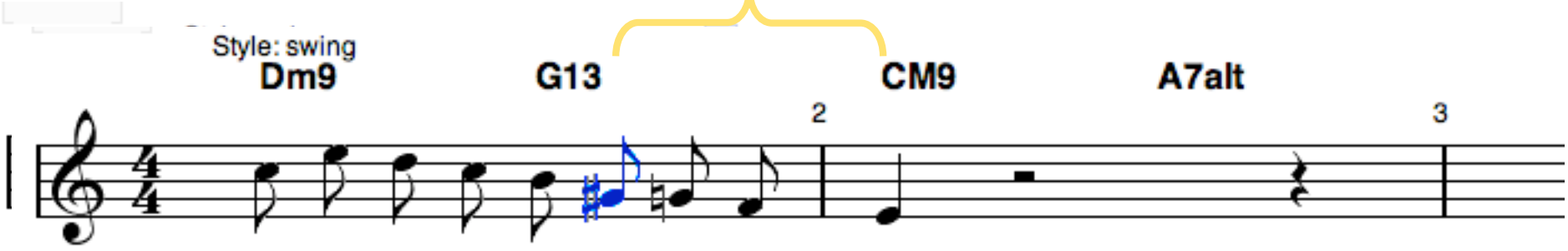


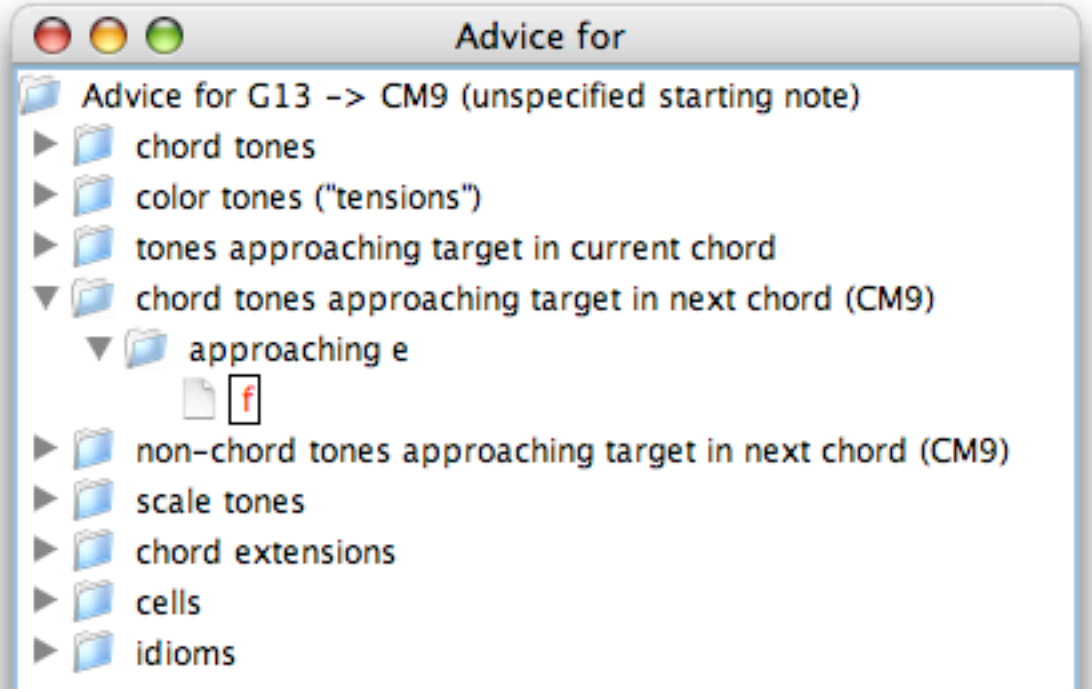
Giving Advice to the User



selected “slot”

Advice Pop-Up:
Things Seasoned Players Know, but Novices Don't

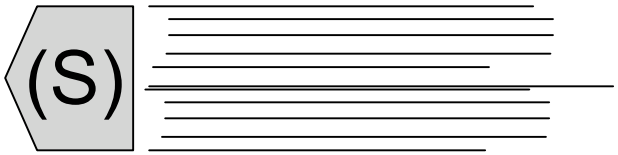




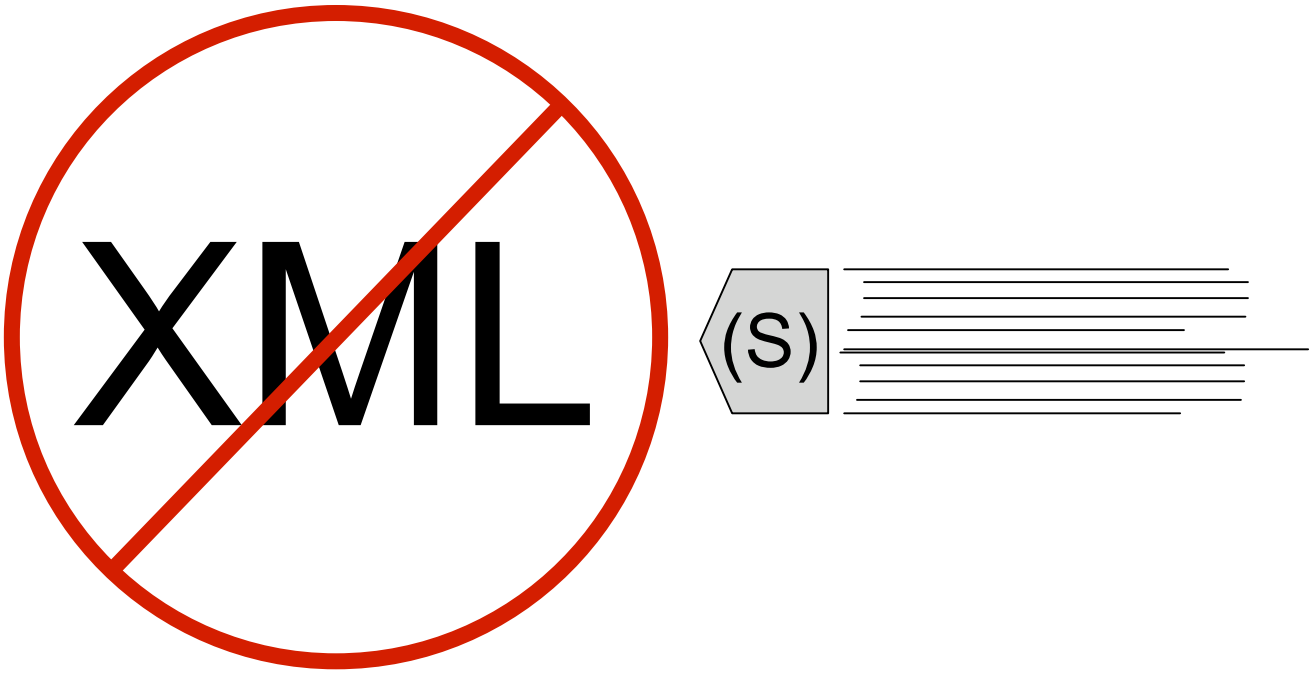
User/Admin Configurability

- User-friendly **S-expression** encodings.
- Several Cohesive Mini-Languages
  - Leadsheet language (for tune or solo)
    Vocabulary language
    - Chord definitions
    - Scale definitions
    - Licks
    - Styles
  - Grammar language
    - For lick generator

There ***was*** a silver bullet.



There ***was*** a silver bullet.

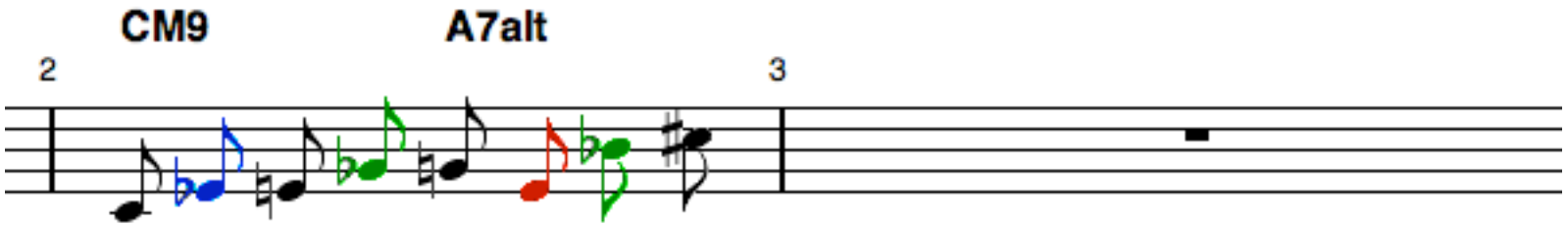


Leadsheet Mini-Language

Designed for Musician's Ease of Use

```
FM69 | Bb13      | FM69       | F#m9 B7       |
Bb13 | Dbm7 Gb7 | FM69       | NC D7alt / / |
Gm9  | C7b9      | F69 D7alt | Gm9 C9        |
r2 a8 r8 c+8 f1
d+8
c+8 bb8 r4 a8 r8 c+8 e2+2/3+16/3
eb4
d2/3+16/3 r8 ab8 c+8 d+8 f+8
e+4 cb8/3 bb8/3 g8/3 gb4 r8 d8
e8 r8 a8 c+2+8
r4 eb+8 c+8 ab8 g8 f#4
r2/3+16/3 d8 f8 a8 c+8 a8
bb4 g8/3 f8/3 e8/3 f8 db2/3+16/3
c2 bb8 c+8 eb+8 c+8
d+8 bb8 g8 f8 e2
```

The Lick Mini-Language



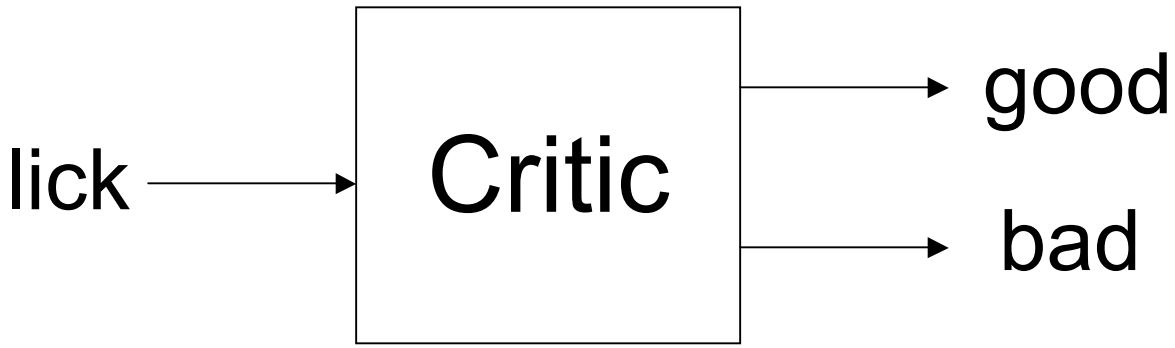
Advice for

- licks
  - with same first chord type (or having this type as an extension)
    - 3.0 beats (2 choices)
    - 3.5 beats (4 choices)
    - 4.0 beats (192 choices)
      - / (1 choice)
      - /\ (22 choices)
      - \/ (6 choices)
      - /\/ (16 choices)
        - /\/ [C -> A7] major 1-6 (b8 c+8 f+8 e+8 b8 bb8 c#+8 e+8)
        - /\/ [C -> A7] major 1-6 (c+8/3 d+8/3 eb+8/3 e+8 c+8 bb8/3 b8/3
        - /\/ [C -> A7] major 1-6 (c+8/3 d+8/3 eb+8/3 e+8 d+8 c#+8/3 bb8
        - /\/ [C -> A7] major 1-6 (c8 eb8 e8 g8 e8 g8 bb8 c#+8)
        - /\/ [C -> A7] major 1-6 (c8 eb8 e8 gb8 g8 e8 bb8 c#+8)

Lick Generation: Machine Learning

- Totally random choices don't sound good.

- Original purpose was to provide examples (good and bad) to ***train*** a classifier (e.g. neural network).



Simple Lick Generator

- Given two-chord sequence and number of beats…
- Specify note probabilities.
- Specify range of durations (e.g eighth-note to quarter-note).
- Specify probability of a rest instead of a note.
- Specify maximum and minimum jump intervals.
- Choose randomly within these parameters.

Results

- For eighth-note only durations, 4-beats, with repetition-avoidance:
  - 90% “useable” licks.
  - Difficult to generate bad examples.
  - Therefore not so good for training classifier.

Results

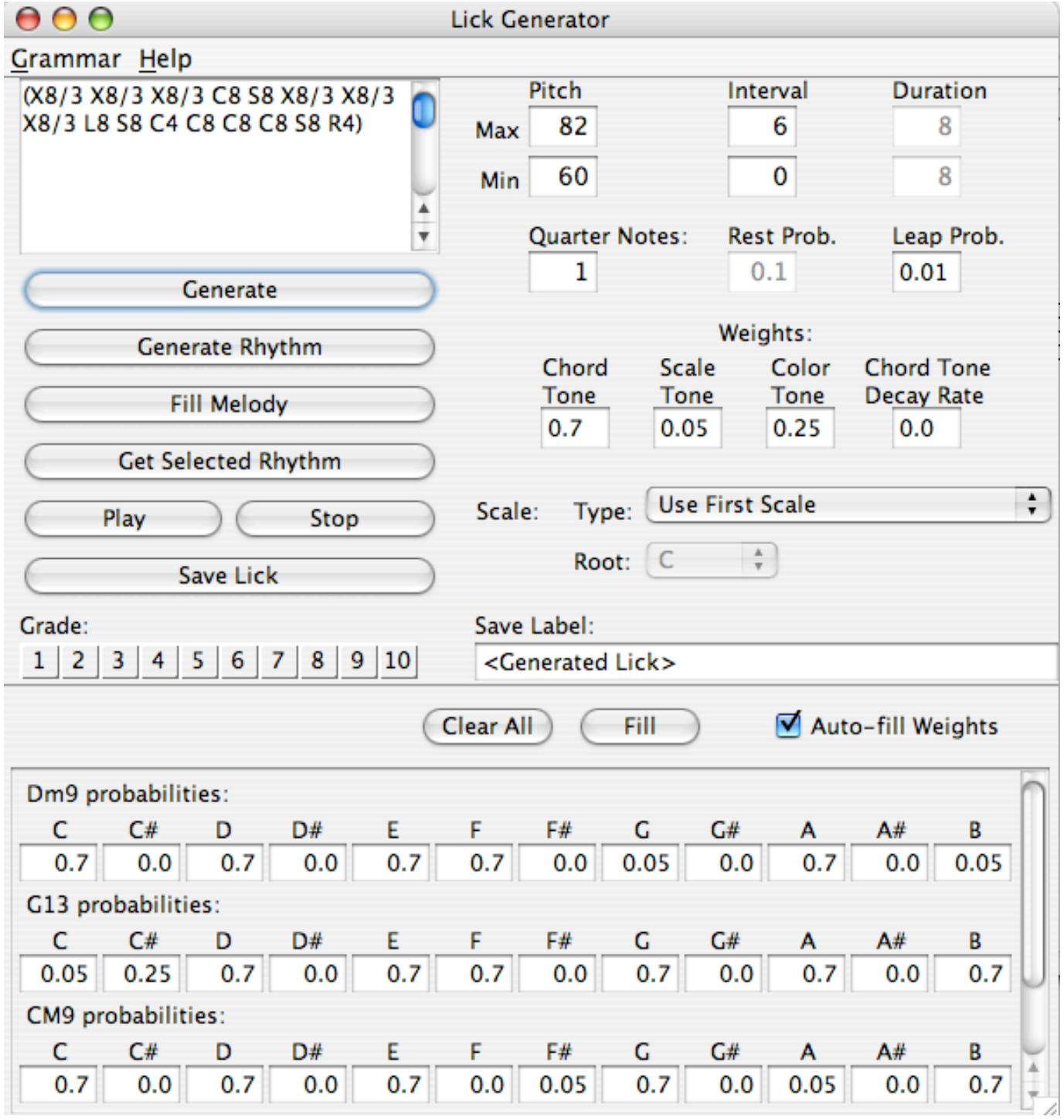
- For eighth-note and quarter note 8 beats, with repetition-avoidance:
  - 60-70% useable
- For wider range of durations and >8 beats, lines often lack coherence due to randomness.

Grammar-Based Lick Generator

- Use probabilistic context-free grammar to generate **annotated rhythms**, then
- Fill in note values probabilistically with choices outlined in grammar.
- Productions humanly-designed to provide coherence.

Grammar Mini-Language

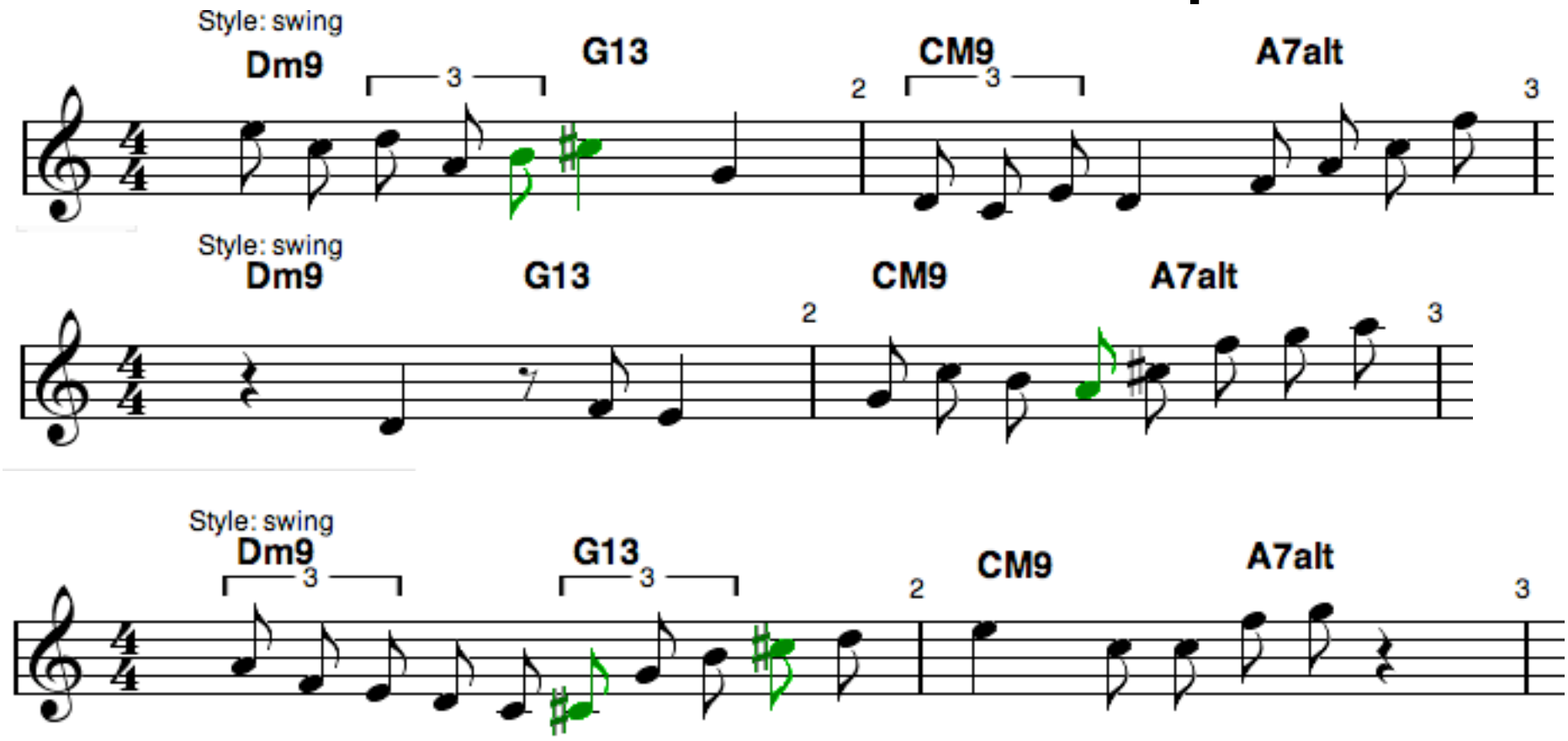
```
(base (P 0) () 1)
(base (P 1) (Seg1) 1)
(base (P 2) (Seg2) 1)
(base (P 3) (Seg2 Seg1) 1)

(rule (P Y) (Seg2 (P (- Y 2))) 0.25)
(rule (P Y) (Seg4 (P (- Y 4))) 0.75)
(rule (Seg4) (Seg2 V4 V4) 0.52)
(rule (Seg4) (V8 N4 N4 N4 V8) 0.01)
(rule (Seg4) (V4 Seg2 V4) 0.47)
(rule (Seg2) (N2) 0.06)
(rule (Seg2) (V4 V4) 0.6)
(rule (Seg2) (V8 N4 V8) 0.12)
(rule (Seg2) (H4. N8) 0.16)
(rule (Seg2) (H4/3 H4/3 H4/3) 0.06)
(rule (Seg1) (C4) 1)
(rule (V4) (N4) 0.22)
(rule (V4) (V8 V8) 0.72)
(rule (V4) (H8/3 H8/3 H8/3) 0.05)
(rule (V4) (H8/3 H8/3 A8/3) 0.01)
(rule (V8) (N8) 0.99)
(rule (V8) (H16 A16) 0.01)
 (rule (N2) (C2) 1)
(rule (N4) (C4) 0.5)
(rule (N4) (L4) 0.2)
(rule (N4) (S4) 0.5)
(rule (N4) (A4) 0.01)
(rule (N4) (R4) 0.25)
(rule (N8) (C8) 0.4)
(rule (N8) (L8) 0.2)
(rule (N8) (S8) 0.4)
(rule (N8) (A8) 0.01)
(rule (N8) (R8) 0.1)
```


Lick Generator
Grammar Help
(X8/3 X8/3 X8/3 C8 S8 X8/3 X8/3 X8/3 L8 S8 C4 C8 C8 C8 S8 R4)
Pitch
Interval
Duration
Max
82
6
8
Min
60
0
8
Quarter Notes:
Rest Prob.
Leap Prob.
1
0.1
0.01
Generate
Generate Rhythm
Fill Melody
Get Selected Rhythm
Play
Stop
Save Lick
Weights:
Chord Tone
Scale Tone
Color Tone
Chord Tone Decay Rate
0.7
0.05
0.25
0.0
Scale:
Type:
Use First Scale
Root:
C
Grade:
1 2 3 4 5 6 7 8 9 10
Save Label:
<Generated Lick>
Clear All
Fill
Auto-fill Weights
Dm9 probabilities:
C C# D D# E F F# G G# A A# B
0.7 0.0 0.7 0.0 0.7 0.7 0.0 0.05 0.0 0.7 0.0 0.05
G13 probabilities:
C C# D D# E F F# G G# A A# B
0.05 0.25 0.7 0.0 0.7 0.7 0.0 0.7 0.0 0.7 0.0 0.7
CM9 probabilities:
C C# D D# E F F# G G# A A# B
0.7 0.0 0.7 0.0 0.7 0.0 0.05 0.7 0.0 0.05 0.0 0.7

Results for Grammar-Based Generator

- Really good (opinion).
- Works over any chord sequence.
- Can construct an entire chorus or multiple choruses.
- Does not use database.
- Tremendous variety: Frequently results in licks not heard before.

Lick Generator Examples



Conjecture

- An implied grammar probably comes closest to what professionals use implicitly (without thinking) in improvising.

Other AI/Language Facets of Impro-Visor

- Chord voicing and voice-leading algorithms
- **Voicing**: how notes of chord are stacked
- **Voice-leading**: smooth flow of notes of one chord into the next

Chord & Voicing Mini-Language

```
(chord
   (name CM69)
   (pronounce C major six nine)
   (family major)
   (spell c e g a d)
   (priority d e a g c)
   (approach (c b c#) (e eb f) (g f# g#) (a g# bb) (d c# eb))
   (color b f#)
   (voicings
      (left-hand-A (type closed) (notes e g a d+))
      (left-hand-B (type closed) (notes g d+ e+ a+))
      (quartal (type open) (notes e a d+ g+))
      (red-garland-A (type hyper-open) (notes e g a d+) (extension d++ g++ d+++))
      (red-garland-B (type hyper-open) (notes g d+ e+ a+) (extension d++ g++ d+++)) )
   (scales
      (C major)
      (C lydian)
      (C bebop major))
   (extensions CM69#11)
   (substitute CM7 CM9 CM69#11) )
```

Bass-Line and Comping Generation

- Automatic creation of bass lines and “comping” according to another set of probabilistic rules.

- Rules are part of **style** mini-language.

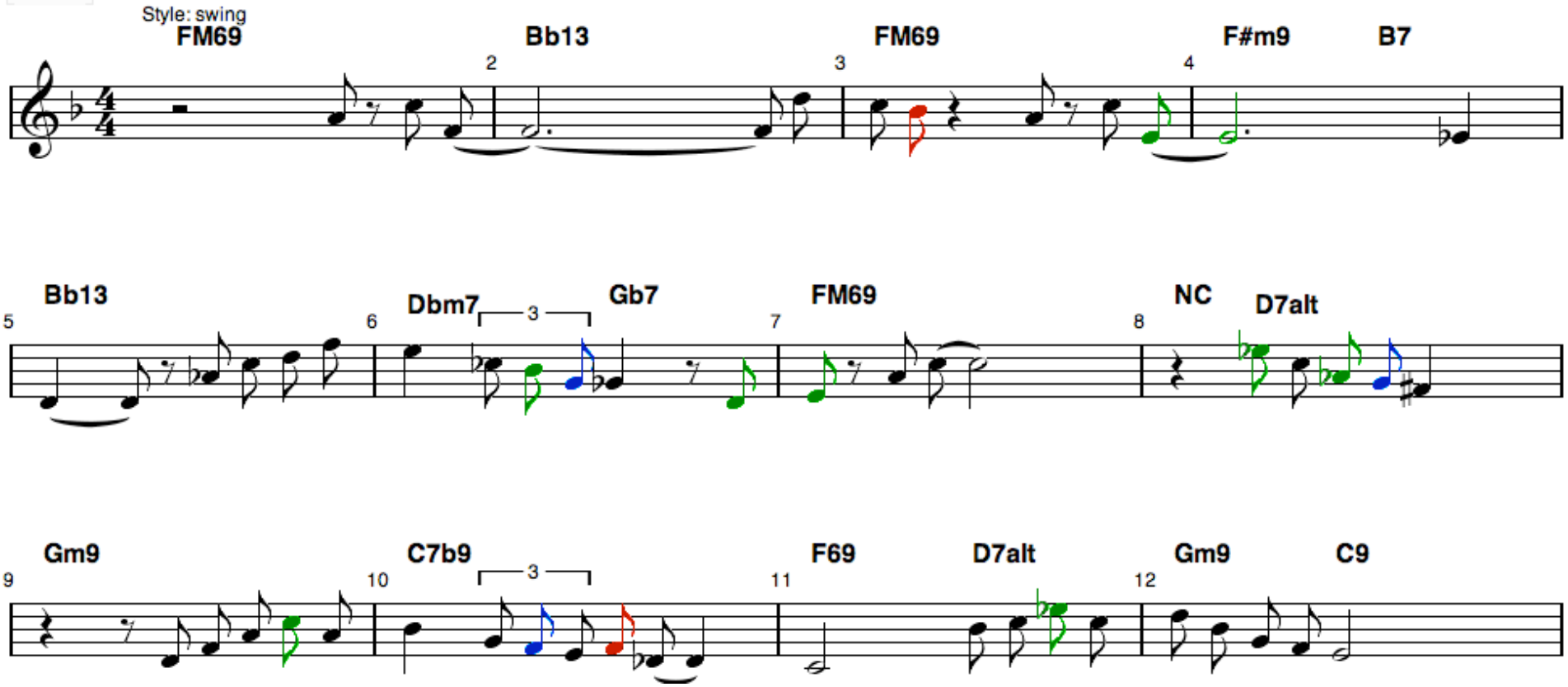
Style Mini-Language

```
(style
    (name swing)
    (swing 0.67)
    (bass-pattern (rules B4 S4 C4 A4) (weight 10))
    (bass-pattern (rules B4 C4 C4 A4) (weight 5))
    (bass-pattern (rules B4 S4 C4 S4) (weight 3))
    (drum-pattern
        (drum 51 X4 X8 X8 X4 X8 X8)
        (weight 10)
        )
    (chord-pattern (rules X1+1) (weight 7))
    (chord-pattern (rules X1) (weight 7))
    (chord-pattern (rules X2) (weight 8)) )
```

Conclusions and Evaluation

- Original purpose of Impro-Visor was to educate users and promote their creativity.
- Successfully used for two semesters by students in a jazz improvisation class.
- Grammar-driven lick generation makes the tool itself creative: thousands of solos without a repeat.
- Other AI aspects present: voicing selection, bass-lines and comping

Future Work

- Further work on critic machine-learning experiment.
- Maybe incorporate critic neural net.
- Non-chordal lick selection.
- Database efficiency.
- Real-time performance.
- Grammar learning.

Blues For Gary