## On-line documentation in the PWMC library

---------------------------------------------------------------------------
build-domain menu
---------------------------------------------------------------------------

MC **BUILD-DOMAIN** (MEASURELAYER NR-SEARCH-VARIABLES RHYTHMLAYER0 PITCHLAYER0 &OPTIONAL RHYTHMLAYER1 PITCHLAYER1 RHYTHMLAYER2 PITCHLAYER2 RHYTHMLAYER3 PITCHLAYER3 RHYTHMLAYER4 PITCHLAYER4 RHYTHMLAYER5 PITCHLAYER5 RHYTHMLAYER6 PITCHLAYER6 RHYTHMLAYER7 PITCHLAYER7 RHYTHMLAYER8 PITCHLAYER8 RHYTHMLAYER9 PITCHLAYER9):
This function takes lists of time signatures, pitches and durations and
formats them into a domain readable by the multi-pmc inside the PWMC library.

Pitches can either be single pitches, single chords, pitch/chord motifs, or transposable pitch motifs (i.e. melodic profiles). Durationscan either be single durations or rhythm motifs.

For more information about the domain, see the PWMC tutorial.

MC **LOCK-RHYTHMS** (RHYTHMSEQ):
Lock the rhythm sequence in one layer to a predefined sequence of durations.

MC **LOCK-PITCHES** (PITCHSEQ):
Lock the pitch sequence in one layer to a predefined sequence of pitches.

MC **RULES->PMC** (&REST RULES):
Collects all rules and formats them to be readable by the multi-pmc.

The box is expandable and accepts any number of rules. Rules can alsobe input as list of rules (the function will make any list flat).

This box needs to be connected to the multi-pmc to make the PWMCsystem work (even if no rule is attached to it, you need to have it attached)!

MC **HEURISTIC-R->PMC** (&REST RULES):
Collects all heuristic rules and formats them to be readable by the multi-pmc inside the PWMC library.

The box is expandable and accepts any number of rules. Rules can alsobe input as list of rules (the function will make any list flat).

MC **MCDECODE** NIL:
This box decodes the output from the multi-pmc and displays it as a score. It is possible to set the tempo in the displayed score (this does not affect the search process in any way).

Three buttons change the behaviour of the score:

all pitches/pitches with rhythms: sets if only pitches with their durations should be displayed, or if pitches that have no duration should be included. In the latter case, these pitches will be printed last, in an extra bar, as whole notes.

all durations/rhythm with pitch: sets if only durations with their pitchesshould be displayed, or if durations that have no pitch should be included.In the latter case, these will be displayed with x-noteheads.

fill sequence/bars from pmc: sets if only music inside bars that where foundby the pmc should be printed, or if the display should provide default time signatures for the music.

MC **PARTIAL-SOLUTION** (TEMPO EXTRA-PITCH? EXTRA-RHYTHM? SUPPLY-MEASURES?):
Get partial solution from last run of PMC.

------------------------------------------------------------------------

Debug tools menu

------------------------------------------------------------------------

MC **MCDEBUG** NIL:
This is a tool to display what happend during the last search. The inputs should not be connected.

The main control is the slider on the top. By dragging the slider to the left, you step back in history from the final solution (or where you interrupted the last search). When you move the slider, the history index displays how far back you are from the found solution (max 100 steps).

The pmc-index shows how far the search process had reached at the displayedpoint.

The score diplays the temporary solution at the history-index. By double clicking the score, you can open a bigger score and set the display the usual way (see the PWGL manual).

MC **DEBUG-ONE-RULE** (FN INDEX):
Test one rule on a variable at index in last solution.

## pitch <-> rhythm rules menu

MC **ACCESS-PITCH-AND-RHYTHM** (TESTFUNCTION LAYERNR MEASURE-INFO? PAUSES? MODE? TYPE?):
This function lets a rule access pairs of pitches and durations in theformat: '(pitch duration)

Layernr can be a list of layernrs - in that case the rule will apply to all layers in the list.

The rule is a testfunction (an abstraction in the lambda state). Each input to the abstraction will access one pitch-duration-pair; two inputs will access two consecutive events (pitch-duration pairs) in the sequence, etc. Any number of inputs is possible.

If an event is a pause, the pitch will be 0 (for example '(0 -1/4)). By filtering out pauses (set the pauses?-menu to exclude_pauses) these event will be skipped.

The mode? menu setting:

Easy mode will give true until all variables are filled with events.

The include-nonexisting-pitches mode will give nil for all variables before the beginning of the sequence. The testfunction has to handle nil-values if this is chosen. Normally this is not a desired option and should be seen as an expert setting.

MC **PITCH->DUR-RULE** (LAYERNR MATCH-DUR? DURATION MATCH-CHORD? CHORDS):
Restricts what pitches are allowed at certain duration lengths.

If the duration at an event is

| | |
|---|---|
| =: | identical to a value |
| member: | member of a list of values |
| longer-than: | longer than a value |
| shorter-than: | shorter than a value |

then the pitch (or chord) at the same event has to be

| | |
|---|---|
| =: | equal to a value |
| member: | member of a list of values |
| lower-than: | lower than a value |
| higher-than: | higher than a value |
| exist: | contain a value (i.e. a chord has to contain a value) |

MC **DUR->PITCH-RULE** (LAYERNR MATCH-CHORD? CHORD MATCH-DUR? DURATIONS):
Restricts what duration lengths are allowed at certain pitches.

If the pitch (or chord) at an event is

=: equal to a value
member: member of a list of values
lower-than: lower than a value
higher-than: higher than a value
exist: contain a value (i.e. a chord has to contain a value)

then the duration at the same event has to be
=: identical to a value
member: member of a list of values
longer-than: longer than a value
shorter-than: shorter than a value

MC **MEL-INTERVAL->DUR-RULE** (LAYERNR MATCH-DUR? DURATIONS MATCH-INTERVAL? INTERVALS):
Restricts what melodic intervals are allowed from an event with a specific duration.

If the duration the melodic interval start at is
=: identical to a value
member: member of a list of values
longer-than: longer than a value
shorter-than: shorter than a value

then the melodic interval has to be
=: equal to an interval
member: member of a list of intervals
smaller-than: smaller than an interval
larger-than: larger than an interval

------------------------------------------------------------------------

meter <-> rhythm rules menu

------------------------------------------------------------------------

MC **ACCESS-METRIC-STRUCTURE** (TESTFUNCTION LAYERNR VARIABLE-TYPE? SELECTION? MODE? TYPE?):
This function lets a rule access durations inside a metric structure.

Layernr can be a list of layernrs - in that case the rule will apply to all layers in the list.

The selection? menu setting:

The at-1st-beat selection will access all durations that start (or exist) at the first beat in each measure. Any duration that does not exist at the first beat will be ignored.

The at-all-beats selection will access all durations that start (or exist) at every beat in each measure. Any duration that does not exist at a beat will be ignored.

The variable-type? menu setting:

The dur_offset will pass durations and their offsets from either the beginning of a beat or the first beat in each measure. The format is '(durtion offset) - for example '(1/4 -1/8) is a quarter note duration starting an eighth note before a beat.

The dur_offs_sign will pass durations, their offsets and the time signature. The format is '(duration offset timesignature) - for example '(1/4 -1/8 (4 4)) is a quarter note duration starting an eighth note before a beat inside a 4//4 bar.

Offset is the distance (in note value) the event has to the beat. 0 means at the beat. -1/8 means an eightnote before the beat. Offsets are always indicated at or BEFORE the beat. The event checked will thus always exist on the beat.

The rule can have any number of inputs/pairs.

MC **METRIC-HIERARCHY-RULE** (LAYERS BEATVALUE0 SUBDIV0 &OPTIONAL BEATVALUE1 SUBDIV1 BEATVALUE2 SUBDIV2 BEATVALUE3 SUBDIV3):
Restricts how rhythm can be put inside a metric framework.

Layers can be a list of layernrs - in that case the rule will apply to alllayers in the list.

Beatvalue is the lower value in a time signature, indicating the length of a beat (i.e. 4 indicating a quarter note length, 8 indicating an eight notelength, etc). The following field will only be applied on measures thathas this beat length.

Subdiv is a list of allowed subdivisions of the beat (the beat is indicatedin the preceeding field). This creates a grid for where onsets are allowed within a measure.

The box can be expanded with additional list of subdivisions of the beat for other beat values.

------------------------------------------------------------------------
rhythm <-> rhythm rules menu
------------------------------------------------------------------------

MC **ACCESS-RHYTHM** (TESTFUNCTION LAYERNR MODE? TYPE?):
This function lets a rule access durations in a layer.

Layernr can be a list of layernrs - in that case the rule will apply to all layers in the list.

The mode? setting:

Easy mode will give true until all variables are filled with durations.

The include-nonexisting-rhythms mode will give nil for all variables before the beginning of the sequence. In this mode the testfunction has to be able to handle nil as a possible value. Normally this is not a desired option and should be seen as an expert setting.

Rhythmcell-easy-mode gives whole rhythmcells (as lists) instead of singledurations for each variable. The rule will be true until all variables are filled with rhythmcells.

The rule (an abstractionm in the lambda state) can have any number of inputs/variables. The inputs correspond to consecutive durations in a layer.

MC **ACCESS-POLY-RHYTHM** (TESTFUNCTION LAYERNR1 LAYERNR2 RANGE? WINDOWSIZE MODE? TYPE?):
This function lets a rule access simultaneous durations in two layers.

The rule (an abstraction in the lambda state) must have only one input.
The range? setting will determine what data the function accesses.

The range? setting:

If range? is set to all-rhythms, the rule will get a list with to sub-lists, each sub-list containing all durations in each layer.

If range? is set to window, the rule will check slices (windows) of the rhythmsequences. The width of the window is determined by the windowsize input. The window will also step forward with the windowsize. The rulewill get access to a list of two sub-list (one for each voice), each sub-list being formated '((duration-seq) offset-from-windowstart).

Ex (window):
'(((1/4 1/4 1/16 1/16) -1/8) ((1/4 1/8) 0))
The first voice start the window with a quarter note, starting 1/8 before the window (i.e. syncopation).
Offsets can only be 0 or negative ratios.

MC **RHYTHMIC-HIERARCHY-RULE** (LAYERHIGH LAYERLOW TYPE?):
Forces a rhythm in one layer to relate in a hierarchical way to a rhythm in another layer.

Layerhigh is the layernr of the higher level in the hierarchy. Onsets in layerhigh have to exist at simultaneous onsets in layerlow. The result is that layerlow reinforces the onsets in layerhigh, however layerlow can have onsets inbetween onsets in layerhigh. Layerlow can be said to have the same rhythm as layerhigh, but with rhythmical ornaments (shorter note values) allowed.

The type? setting:

all_dur: All durations are checked regardless of if the domain defines
rhythms as motifs or not. Pauses are ignored.

low-only-cellstart: The lower layer in the hierarchy has to have its cellstarts align at the time points for the onsets of layerhigh. Pauses as cellstarts are not valid points in the hierarchy, i.e. these cells can not be used at these points.

MC **RHYTHMIC-CANON-RULE** (LAYERDUX LAYERCOMES OFFSET FREE-START? &OPTIONAL SCALETIME):
Forces the rhythm in one layer to imitate the rhythm in another layeras a strict canon.

layerdux: This is the voice that starts the canon.
layercomes: This is the voice that follows layerdux.
offset: This sets how much later the imitation in layercomes starts.

The free-start? setting:

pause: This forces layercomes to have pause(s) until the imitation starts.
If this option is selected the domain has to contain the necessary pausesfor this offset to be possible.

any-rhythm: This allows layercomes to have any rhythm (or pause) before the
imitation starts.

Optional a factor (scaletime) determines if the durations in the imitation should be identical to the original voice (scaletime = 1, default) or if they should be scaled by the given factor. For example if scaletime is set to 2, the imitation will be in double note values.

------------------------------------------------------------------------

pitch <-> pitch rules menu

------------------------------------------------------------------------

MC **ACCESS-MELODY** (TESTFUNCTION LAYERNR MODE? TYPE?):
This function lets a rle access pitches in a layer.

Layernr can be a list of layernrs - in that case the rule will apply to alllayers in the list.

The mode? setting:

Easy-mode will give true until all variables are filled with pitches

The include-nonexisting-pitches mode will give nil for all variables before the beginning of the sequence. In this mode the testfunction has to be able to handle nil as a possible value. Normally this is not a desired option and should be seen as an expert setting.

List-all-pitches gives a list with all pitches that exist in a layer so far. If this setting is chosen the rule must only have one input.

If easy-mode or include-nonexisting-pitches is chosen, the rule (an abstraction in the lambda state) can have any number of inputs/variables. The inputs correspond to consecutive durations in a layer.

MC **ACCESS-2PART-HARMONY** (TESTFUNCTION LAYERNR1 LAYERNR2 SELECTION? HEUR? &OPTIONAL BEAT-SUBDIV):
This function lets a rule access simultaneous pitches in 2 layers.

The rule (an abstractionm in the lambda state) can have any number ofinputs/variables. Two or more inputs represent consecutive harmonies (see the selection? setting below). The test function is only called when both pitches exist. If a pause appears, the test is ignored.

The selection? setting:

always: The rule is checked when a new pitch starts in any of the two voice.

homophony: The rule is only checked for pitches that exist at homophonic rhythms (i.e. they must have simultaneous onsets). Consecutive harmonies are here harmonies that exist on consecutive homophonic events.

onset-on-beat: The rule is only checked for pitches that starts at metric beats (= the onset in any of the voices has to occur at a beat). Consecutive harmonies are here harmonies that exist on consecutive beats. The optional beat-subdiv input determines if the rule should be checked on subdivisions of beats as well (for example 2 = every beat is divided into two equal parts, and each onset that coincides with any of these time points is checked).

duration-on-beat: The rule is checked for pitches that exist at metric beats (i.e. regardless of where the onsets are). The optional beat-subdiv setting works as above.

1-st-voice: The rule is checked at the time points where the onsets for the first voice are.

MC **RULE-4-VOICE-CHORDS** (CHORD-LIST LAYERNR1 LAYERNR2 LAYERNR3 LAYERNR4 SELECTIONS? HEUR? &OPTIONAL BEAT-SUBDIV):
Makes sure that chord structures between up to 4 voices match a list of allowed chord structures. Only pitch classes for the pitches are checked (i.e. pitches can be in any octave).

chord-list: This is the list of allowed chord structures. Numbers indicate intervals from the lowest pitch in the definition. Chords are allowed in any tranposition and position. Ex. (4 7) is a major triad. The rule will allow (3 8) and (5 9) as well since they are also major triads built form the third respective fifth.

Layernrs determines the layers the pitches in the chord exist in. The pitches have to be single pitches (chords within a voice will not be understood). By setting two or more layernrs to the same number, the rule can also handle chords/intervals between 2 or 3 voices (unisons are always allowed).

The selection? setting:

always: The rule is checked when a new pitch starts in any of the voices.

duration-on-beat: The rule is only checked for pitches that exist at metric beats (i.e. regardless of where the onsets are). The optional beat-subdiv input determines if the rule should be checkes on subdivisions of the beat as well (for example 2 = every beat is divided into two equal parts, and the starting point of each subdivision is checked).

MC **PITCH-CANON-RULE** (LAYERNR1 LAYERNR2 &OPTIONAL TRANSPOSE-LAYER2):
Forces the pitches in one layer to imitate the pitches in another layer in a strict canon. If a traditional canon will be perceived depends on the durations.

Optional the second layer can be transposed by the interval defined in the transpose-layer2 setting.

MC **STARTPITCH-RULE** (LAYERNRS PITCHES):
Forces the first pitch in a layer to be fixed to a given value.

layernrs: This is a list of the layernrs for the layers
pitches: This is a list of the first pitch for each layer in layernrs.

The lists in layernrs and pitches have to be of the same lengths.

MC **ACCESS-HARMONY** (TESTFUNCTION LAYERNR1 LAYERNR2 RULE HEUR? &OPTIONAL BEAT-SUBDIV):

The rule (an abstractionm in the lambda state) must have two inputs/variables. The first is the first layer's pitch, the second is the second layer's simultaneous pitch. The test function is only called when both pitches exist. If a pause appears, the test is ignored.

The rule menu determines how the test function is checked:

Always - The test function is checked for each separate pitch. Homophony - The test function is checked for pitches that exist in homophonic rhythm (onsets).

Onset-on-beat - The testfunction is checked for pitches that start at metric beats (= onset exist at metric beat). The optional beat-subdiv input determines if the testfunction should be checked on subdivisions of beats as well (2 = every beat is divided into two equal divisions, and each onsets that occur on any division is checked).
Duration-on-beat - The testfunction is checked for pitches that exist at metric beats (= metric beats occurs within the duration of the pitch).

Layernr can be a list of layernrs - in that case the rule will apply to all layers.

## meter <-> meter rules menu

MC **ACCESS-METER** (TESTFUNCTION MODE? TYPE?):
This function lets a rule access time signatures. If the rule has more than one input, time signatures in cosecutive measures will be checked.

MC **LOCK-METER-RULE** (TIMESIGNS &OPTIONAL NR-OF-MEAURES):
This rule will lock the sequence of timesigns to be equal to a predefined sequence of timesigns. The rule will not affect measures after the length of the predefined sequence.

## strategy rules menu

MC **STRATEGY-RULE-1LAYER** (LAYERNR STRATEGY &OPTIONAL OFFSET):
Strategy rule within a layer. Layernr can be a list of layers - the rule will then be applied on all layers in the list.

The strategy rule controls what should be assigned first (for example pitch or duration). A good strategy will speed up a search. You need to think about if it makes sense to for example put pitches on existing rhythms or vice versa.

The equal length settings will not prefere any to be assigned first, but will force the search to assign what is missing (i.e. if durations exist without pitch, then find pitches).

The before-setting will first assign what you have chosen to be first, but then immediately fill in the missing parameters (unless an offset is used).

IMPORTANT ABOUT OFFSETS:
pitch-before-rhythm and rhythm-before-pitch: offset (optional) is in number of notes.

mea-before-rhy and rhy-before-mea: offset is in duration (note value)

MC **STRATEGY-RULE-2LAYERS** (LAYERNR1 LAYERNR2 STRATEGY &OPTIONAL OFFSET):
Strategy rule between layers.

The strategy rule controls what should be assigned first (for example pitch in layer 1 before pitch in layer 2). A good strategy will speed up a search. You need to think about if it makes sense to for example searh for pitches in layer 2 when pitches in layer 1 exist or vice versa.

The equal length settings will not prefere any to be assigned first, but will force the search to assign what is missing (i.e. if 5 pitches in layer 1 exist, then find 5 pitches in layer 2).

The before-setting will first assign what you have chosen to be first, but then immediately fill in the missing items (unless an offset is used).

IMPORTANT ABOUT OFFSETS:
pitch-before-pitch: offset (optional) is in number of pitches
rhythm-before-rhythm: offset is in duration (note value)

MC **STRATEGY-ONLY-MOTIFS** (LAYERNR):
The result of this strategy rule is that the first pitches will be a from single pitch-cell with fixed pitches from the domain, and all following pitches will be determined from transposable pitch cells.

------------------------------------------------------------------------
freeze menu
------------------------------------------------------------------------

MC **STORE-SOL** (SOLUTION-FROM-PMC):
Store part of a solution to freeze in next calculation.
The solution to store should come directly from the pmc. Use this function with freeze-rule

MC **FREEZE-RULE** (END-INDEX DATA):
Freeze the first n variables from last solution in next calculation. Use this rule with store-sol.

MC **NR-OF-EVENTS** NIL:
Get number of events (i.e. number of pitches OR number of note values and pauses OR number of time signatures) including the one being examined.