

// This Pine Script® code is subject to the terms of the Mozilla Public License 2.0 at
<https://mozilla.org/MPL/2.0/>

// © LuxAlgo

//@version=6

indicator('MFB - Maximization Prophet', overlay = true)

// =====

// 🕒 High Probability Trading Windows (EST)

// =====

grp_win = ' 🕒 Trade Windows (EST)'

manage_all = input.bool(true, 'Manage Signals Outside Windows', group = grp_win, tooltip =
'If checked, the Prophet will manage signals 24/7. If unchecked, it only activates during
your High Probability Windows.')

win1 = input.bool(true, '1) 07:30 - 09:00 (NY Volume)', group = grp_win)

win2 = input.bool(true, '2) 02:00 - 05:00 (London Open)', group = grp_win)

win3 = input.bool(true, '3) 10:00 - 11:00 (London Close)', group = grp_win)

win4 = input.bool(true, '4) 13:30 - 15:00 (Afternoon Repo)', group = grp_win)

win5 = input.bool(true, '5) 18:00 - 00:00 (Sunday Gap)', group = grp_win)

win6 = input.bool(true, '6) 09:30 - 09:45 (NY Opening Range)', group = grp_win)

is_in_window() =>

t1 = not na(time(timeframe.period, '0730-0900', 'America/New_York')) and win1

t2 = not na(time(timeframe.period, '0200-0500', 'America/New_York')) and win2

t3 = not na(time(timeframe.period, '1000-1100', 'America/New_York')) and win3

t4 = not na(time(timeframe.period, '1330-1500', 'America/New_York')) and win4

```
t5 = not na(time(timeframe.period, '1800-0000', 'America/New_York')) and win5 and
dayofweek == dayofweek.sunday
```

```
t6 = not na(time(timeframe.period, '0930-0945', 'America/New_York')) and win6
manage_all or (t1 or t2 or t3 or t4 or t5 or t6)
```

```
bgcolor(not manage_all and is_in_window() ? color.new(#5b9cf6, 95) : na)
```

```
// =====
```

```
// 🟡 Risk Settings
```

```
// =====
```

```
grp_sl = ' 🟡 Risk Settings'
```

```
sl_offset = input.float(4.0, 'Initial SL Offset (Points)', minval = 0, group = grp_sl, tooltip =
'Rule: 5m candle extremity +/- 3 to 5 points.')
```

```
use_be = input.bool(true, 'Move to BE on B.O.S.', group = grp_sl, tooltip = 'Rule: Move stop
to Break-Even when a Break of Near Term Structure occurs.')
```

```
ext_right = input.int(20, 'Extend SL Right (Bars)', minval = 1, group = grp_sl)
```

```
// =====
```

```
// 🇺🇸 Rule Detection Logic
```

```
// =====
```

```
// 1. Structural Pivots (B.O.S & Key Levels)
```

```
ph = ta.pivohigh(high, 3, 3)
```

```
pl = ta.pivotlow(low, 3, 3)
```

```
var float last_ph = na, var float last_pl = na
```

```
if not na(ph)
```

```
    last_ph := ph
```

```
if not na(pl)
```

```
last_pl := pl
```

```
// 2. FVG (Protected FVGs)
```

```
fvg_bull = low > high[2]
```

```
fvg_bear = high < low[2]
```

```
// 3. Signal Aggregation
```

```
uni_bull_signal = ta.crossover(close, last_ph) and is_in_window()
```

```
uni_bear_signal = ta.crossunder(close, last_pl) and is_in_window()
```

```
buy_signal = uni_bull_signal or (fvg_bull and is_in_window()) and close > high[2]
```

```
sell_signal = uni_bear_signal or (fvg_bear and is_in_window()) and close < low[2]
```

```
// =====
```

```
// 🟡 Rule-Based Management (The Prophet)
```

```
// =====
```

```
// Initial 5m Closure Context
```

```
[h5, l5] = request.security(syminfo.tickerid, '5', [high, low], lookahead =  
barmerge.lookahead_on)
```

```
h5_safe = nz(h5, high), l5_safe = nz(l5, low)
```

```
var float sl_line = na
```

```
var float active_entry = na
```

```
var bool is_long = false
```

```
var bool at_be = false
```

```
// --- Rule 1: Trade Activation (WHERE/WHEN) ---
```

```
if buy_signal and na(sl_line)
```

```
    is_long := true
```

```
    active_entry := close
```

```
    sl_line := l5_safe - (sl_offset * syminfo.pointvalue)
```

```
    at_be := false
```

```
if sell_signal and na(sl_line)
```

```
    is_long := false
```

```
    active_entry := close
```

```
    sl_line := h5_safe + (sl_offset * syminfo.pointvalue)
```

```
    at_be := false
```

```
// --- Rule 2-4: Ongoing Risk Management ---
```

```
if not na(sl_line)
```

```
    // Rule 2: Move to BE on B.O.S.
```

```
    bos_occ = is_long ? ta.crossover(close, last_ph) : ta.crossunder(close, last_pl)
```

```
    if use_be and bos_occ and not at_be
```

```
        sl_line := active_entry
```

```
        at_be := true
```

```
// Rule 3: Trail Protected FVGs (Retrace and Respect)
```

```
if is_long and fvg_bull and high[2] > sl_line
```

```
    sl_line := high[2]
```

```
else if not is_long and fvg_bear and low[2] < sl_line
```

```
    sl_line := low[2]
```

```

// Rule 4: Adjust on Key Level Hit (Pivot)

key_hit = is_long ? high >= last_ph : low <= last_pl

if key_hit

    tighten = is_long ? low[1] : high[1]

    if (is_long and tighten > sl_line) or (not is_long and tighten < sl_line)

        sl_line := tighten


// --- Exit Condition (Rule Integrity Check) ---

if not na(sl_line)

    if (is_long and low <= sl_line) or (not is_long and high >= sl_line)

        sl_line := na

        active_entry := na


// =====

// 🎨 Final Visuals

// =====

// Clean Signal Markers

plotshape(buy_signal and na(sl_line[1]), "Entry Long", shape.triangleup, location.belowbar,
color.green, 0, size = size.small)

plotshape(sell_signal and na(sl_line[1]), "Entry Short", shape.triangledown,
location.abovebar, color.red, 0, size = size.small)


// THE MAXIMIZATION PROPHET (Horizontal Current Stop Only)

var line sl_ext = line.new(na, na, na, na, color=#f23645, width=1, style=line.style_solid)

var label sl_lbl = label.new(na, na, "", color=#f23645, textcolor=color.white,
style=label.style_label_left, size=size.small)

```

```
if barstate.islast and not na(sl_line)

    line.set_xy1(sl_ext, bar_index, sl_line)

    line.set_xy2(sl_ext, bar_index + ext_right, sl_line)

    label.set_xy(sl_lbl, bar_index + ext_right, sl_line)

    label.set_text(sl_lbl, " SL: " + str.tostring(sl_line) + " ")

else

    line.set_xy1(sl_ext, na, na)

    label.set_xy(sl_lbl, na, na)


// Suble Entry Reference

plot(active_entry, "Entry Level", color.new(color.gray, 80), 1, plot.style_linebr)
```