Имя: Библиотека функций и советники для трейлинга / Юрий Дзюбан

Автор: I\_D (13.09.2007 17:54)

Рейтинг: 9.8

Скачано: 4557

Скачать: TrailingExperts.zip (74.1 Kb)

MyFractals(example).mq4 (8.2 Kb) View

TrailingFuncLib.mq4 (58.4 Kb) View

Алгоритм сопровождения и закрытия позиции, как известно, является едва ли не более важным, чем правила её открытия. И подчас изменение системы трейлинга может вызывать довольно драматические перемены в результативности торговли - как в сторону повышения, так и понижения последней. Предлагаю Вашему вниманию библиотеку функций для сопровождения открытой позиции, которую я использую при построении своих МТС.

Представленные функции были написаны по некоторым наиболее популярным алгоритмам трейлинга, которые я в разное время встречал в сети, другие являются, возможно, моими авторскими разработками или, по крайней мере, мне не попадались их описания до момента написания. Функции просто подключать/перебирать, что позволяет весьма оперативно их примерять к системе входов различные сценарии трейлинга. Представленные функции являются элементарными и позволяют, при необходимости, конструировать на их основе более сложные алгоритмы трейлинга. Функции весьма разные, писались мной для упрощения процедуры сборки новых МТС.

Итак, собственно функции:

1. Трейлинг по теням N последних баров.

void TrailingByShadows(int ticket,int tmfrm,int bars\_n,

int indent,bool trlinloss)

Функция реализует подтягивание стоплосса по минимальному из low (для длинных позиций) или максимальному high (для коротких) заданного количества баров, на текущем или отличном от текущего таймфрейме.

Параметры:

ticket - уникальный порядковый номер ордера (выбранный перед вызовом функции с помощью OrderSelect());

tmfrm - таймфрейм, по барам которого осуществляется трейлинг (варианты - 1, 5, 15, 30, 60, 240, 1440, 10080, 43200);

bars\_n - количество баров, которые используются для определения уровня стоплосса (не меньше 1);

indent - отступ (пунктов) от выбранного high/low, на котором будет размещен стоплосс (не меньше 0);

trlinloss - указатель того, следует ли передвигать стоплосс на "лоссовом" участке, т.е. в интервале между начальным стоплоссом и курсом открытия (true - тралим, false - трейлинг начинается только при условии, что новый стоплосс "лучше" курса открытия, "в профите").

Данный вид трейлинга был впервые встречен мной в описании тактики В. Баришпольца "Пробой средней на дневках", в которой предлагалось трейлинговать стоплосс по лоу (для покупок) или хай (продаж) последних 2 баров (свечей) на дневном таймфрейме. Тралит данный алгоритм иногда на удивление "красиво" (попробуйте "пройтись" по графику "глазами"), позволяет переживать откаты (глубина зависит от заданного количества баров, bars\_n) и периоды ослабления волатильности. (По определению) срабатывает по закрытию очередного бара, даже если вы используете его на каждом тике. Я использую его преимущественно на дневках, тралю по 2-4 (наиболее часто) последним барам.

2. Трейлинг по Х-барным фракталам.

void TrailingByFractals(int ticket,int tmfrm,int frktl\_bars,

int indent,bool trlinloss)

Функция тралит стоплосс по экстремумам фракталов заданого размера (т.е. кол-ва баров в составе фрактала) на заданном таймфрейме. Фрактал, как известно, в "классическом" виде (Вильямса, если не ошибаюсь) - это комбинация из 5 баров, но мне подумалось, а почему бы не использовать и другие "размеры" - начиная от 3 баров и выше, включая парные кол-ва баров (в последнем случае кол-во баров после экстремума, по мере движения "в прошлое", будет больше, чем до экстремума).

Параметры:

ticket - уникальный порядковый номер ордера(выбранный перед вызовом функции с помощью OrderSelect());

tmfrm - таймфрейм, по барам которого осуществляется трейлинг (варианты - 1, 5, 15, 30, 60, 240, 1440, 10080, 43200);

bars\_n - количество баров в составе фрактала (не меньше 3);

indent - отступ (пунктов) от экстремума последнего фрактала, на котором будет размещен стоплосс (не меньше 0);

trlinloss - указатель того, следует ли передвигать стоплосс на "лоссовом" участке, т.е. в интервале между начальным стоплоссом и курсом открытия (true - тралим, false - трейлинг начинается только при условии, что новый стоплосс "лучше" курса открытия, "в профите").

Не раз встречал на нескольких из основных рунетовских форумов Форекс-тематики утверждения относительно того, что "хорош трейлинг по фракталам на более низком таймфрейме". Поскольку фракталы, по сути, обозначают локальные экстремумы, трейлинг по ним, также, в принципе, можно назвать трейлингом по локальным экстремумам. Идея не новая и вполне здравая, использовавшаяся многими изветными трейдерами (тем же Вильямсом, если не ошибаюсь). Позволяет переживать откаты и неплохо "выбирать" заметные движения, неважно (по личному опыту) работает во флэте. Подобно предыдущей, данная функция "тралит" по сформировавшимся барам. При желании можно динамически (в пределах "жизни" позиции) изменять параметры, например, при быстром "приросте" профите (например, при сильном тренде), "загрублять" её чувствительность, переходя на более крупный таймфрейм или увеличивая кол-во баров во фрактале, или, наоборот, при спадах волатильности повышать чувствительность, переходя на более мелкий таймфрейм или уменьшая кол-во баров во фрактале. Возможны и другие варианты.

3. Трейлинг стандартный-'ступенчатый'.

void TrailingStairs(int ticket,int trldistance,int trlstep)

Данный вид трейлинга является доработкой стандартного. Если не ошибаюсь, в своё время похожий был написан KimIV'ом (но поскольку "свое - роднее",... :) Отличается от стандартного трейлинга тем, что перенос стоплосса осуществляется не "попунктово" (например, при трейлинге на расстоянии 30 пунктов при +31 стоплосс будет пенемещен на +1, при +32 - на +2 и т.д.), а "шагами" заданной длины (например, при трейлинге на расстоянии в 40 пунктов и "шаге" в 10 при достижении +40 стоплосс будет перемещен на +10, дальше до момента достижения профитом +50 (40 п. + шаг) ничего не поменяется (т.о. мы даём цене определённую свободу, в чем, собственно, и суть данного алгоритма), и только при +50 стоплосс с +10 будет перенесен на шаг - с +10 на +20, при +60 - на +30 п. и т.д.

Параметры:

ticket - уникальный порядковый номер ордера (выбранный перед вызовом функции с помощью OrderSelect());

trldistance - расстояние от текущего курса (пунктов), на котором "тралим" (не меньше MarketInfo(Symbol(),MODE\_STOPLEVEL));

trlstep - "шаг" изменения стоплосса (пунктов) (не меньше 1).

При trlstep=1 данная функция не будет отличаться от стандартного трейлинга. Основная "фишка" данного алгоритма, повторюсь, в предоставлении курсу некоторой "свободы движений" - стоплосс подтягивается только после того, как цена "побродив, определилась". Данный алгоритм трейлинга я впервые встретил в описании правил тактики "Скользящие каналы" уже упоминавшегося В.Баришпольца.

4. Трейлинг стандартный-'удавка'.

void TrailingUdavka(int ticket,int trl\_dist\_1,int level\_1,

int trl\_dist\_2,int level\_2,int trl\_dist\_3)

Безоткатных движений неограниченной длины, как известно, не бывает (эх...). И совершив определённый "рывок" (среднее значение которого можно попытаться определить, например, статистически), курс, как правило, формирует консолидацию или откат. Идея данного алгоритма сопровождения - по мере увеличения профита сокращать дистанцию трейлинга таким образом, чтобы подходя к участку вероятного разворота/консолидации "застолбить" побольше профита. Помимо работы на упомянутых выше "рывках" (которые обычно имеют место на новостях или при пробое исторических уровней), функцию также можно использовать при работе в каналах, сокращая дистанцию трейлинга при подходе к противоположной (относительно открытия) границе канала.

Параметры:

ticket - уникальный порядковый номер ордера (выбранный перед вызовом функции с помощью OrderSelect());

trl\_dist\_1 - исходное расстояние трейлинга (пунктов) (не меньше MarketInfo(Symbol(),MODE\_STOPLEVEL), больше trl\_dist\_2 и trl\_dist\_3);

level\_1 - уровень профита (пунктов), при достижении которого дистанция трейлинга будет сокращена с trl\_dist\_1 до trl\_dist\_2 (меньше level\_2; больше trl\_dist\_1);

trl dist\_2 - расстояние трейлинга (пунктов) после достижения курсом уровня профита в level\_1 пунктов (не меньше MarketInfo(Symbol(),MODE\_STOPLEVEL));

level\_2 - уровень профита (пунктов), при достижении которого дистанция трейлинга будет сокращена с trl\_dist\_2 до trl\_dist\_3 пунктов (больше trl\_dist\_1 и больше level\_1);

trl dist\_3 - расстояние трейлинга (пунктов) после достижения курсом уровня профита в level\_2 пунктов (не меньше MarketInfo(Symbol(),MODE\_STOPLEVEL)).

К примеру, при параметрах "ticket#, 30, 50, 20, 70, 10" мы исходно тралим на расстоянии в 30 пунктов, как только курс отойдет от курса открытия на 50 пунктов, начинаем тралить "плотнее" - на расстоянии в 20 п., а если ему удалось преодолеть 70 п. в направлении увеличения профита, "затягиваем" (откуда и название) расстояние трала до 10 п. (что с высокой вероятностью приведет к скорому закрытию позиции).

5. Трейлинг по времени.

void TrailingByTime(int ticket,int interval,int trlstep,

bool trlinloss)

Тралим по времени, безотносительно к текущему результату по позиции и раскладу на рынке. Через заданные интервалы времени (целых минут) пытаемся (по возможности) переместить стоплосс на заданный шаг. Использование данного трейлинга отдельно не было (для меня, по крайней мере, возможно, для кого-то окажется иначе) особо успешным, однако в комбинации с другими видами трала или при включении по определённому условию, возможно, он может быть полезен.

Параметры:

ticket - уникальный порядковый номер ордера (выбранный перед вызовом функции с помощью OrderSelect());

interval - количество целых минут с момента открытия позиции, по истечению которых пытаемся переместить стоплосс на шаг trlstep пунктов;

trlstep - шаг (пунктов), на который пытаемся перемещать стоплосс через каждые interval минут;

trlinloss - в данном случае если trlinloss==true, то тралим от стоплосса, иначе от курса открытия (если стоплосс не был установлен, ==0, тралим всегда от курса открытия).

6. Трейлинг по ATR.

void TrailingByATR(int ticket,int atr\_timeframe,int atr1\_period,

int atr1\_shift,int atr2\_period,int atr2\_shift,

double coeff,bool trlinloss)

ATR - (Average True Range, Средний Истинный Диапазон) - один из индикаторов волатильности; чем больше значение, тем, соответственно, выше средняя волатильность за заданный период (индикатора) времени; измерятеся в пунктах. Трейлинг по ATR в большинстве случаев позволяет изменять стоплосс соответственно ("адекватно", "адаптивно") характеру поведения курса - при высокой волатильности (выраженных рывках) курс "отпускаем", при "топтании на месте" поджимаем более "плотно". В трейлинге используется 2 ATR, которым предлагается задавать отличные периоды - один короткий (напр., 5), другой - длинный (напр., 20). Для расчета стоплосса используется всегда большее из значений 2 ATR - это сделано для того, чтобы несколько низковолатильных баров подряд (например, перед выходом новостей) не сделали наш стоплосс слишком близким к текущему курсу.

Параметры:

ticket - уникальный порядковый номер ордера (выбранный перед вызовом функции с помощью OrderSelect());

atr\_timeframe - таймфрейм, на котором рассчитываем значение ATR (варианты - 1, 5, 15, 30, 60, 240, 1440, 10080, 43200);

atr1\_period - период первого ATR (больше 0; может быть равен atr2\_period, но лучше отличен от последнего, почему - см. выше);

atr1\_shift - для первого ATR сдвиг "окна", в котором рассчитывается значение ATR, относительно текущего бара на указанное количество баров назад (неотрицательное целое число);

atr2\_period - период второго ATR (больше 0);

atr2\_shift - для второго ATR сдвиг "окна", в котором рассчитывается значение ATR, относительно текущего бара на указанное количество баров назад (неотрицательное целое число);

coeff - стоплосс считаем как ATR\*coeff, т.е. это коэффициент, определяющий, на расстоянии скольких ATR от текущего курса следует разместить стоплосс;

trlinloss - указатель того, следует ли передвигать стоплосс на "лоссовом" участке, т.е. в интервале между начальным стоплоссом и курсом открытия (true - тралим, false - трейлинг начинается только при условии, что новый стоплосс "лучше" курса открытия, "в профите").

Также довольно популярный способ трейлинга, (в большинстве случаев выгодно) отличающийся от стандартного "адаптивностью" к текущей ситуации на рынке.

7. 'Храповой' трейлинг (по Баришпольцу).

void TrailingRatchetB(int ticket,int pf\_level\_1,int pf\_level\_2,

int pf\_level\_3,int ls\_level\_1,

int ls\_level\_2,int ls\_level\_3,

bool trlinloss)

Храповой (англ. ratchet) ключ, колесо - тот, который "проворачивается только в одном направлении". Название - моё, в описании Баришпольца упоминается как "вторая тактика трейлинга" (в противовес "первой", со "стандартными" - дальними стоплоссами), в исполнении KimIV - т.н. "трехуровневый" трейлинг. Суть: быстрый перенос в безубыток и ступенчатый трейлинг при небольшом профите (дальше предлагается применять иные виды трейлинга, например, стандартный, можно мой стандартный-"ступенчастый" или какой-нибудь другой из вышеописанных). Целесообразность такого трала - "абсолютное большинство позиций по крайней мере некоторое время дают профит", поэтому при первой возможности переносим стоплосс в безубыток; если преждевременно "закроет", но сигналы к открытию сохраняются, возможен повторный вход; а вообще получаем большое кол-во сделок, закрытых в безубыток, и несколько, возможно, с хорошим профитом - на тех сделках, где после открытия последовало сильное движение в профитном направлении. Автор подчеркивает, что данный трейлинг не является "пипсовочным", а предназначен для достижения торговли с низким % убыточных сделок.

Параметры:

ticket - уникальный порядковый номер ордера (выбранный перед вызовом функции с помощью OrderSelect());

pf\_level\_1 - уровень профита (пунктов), при котором стоплосс переносим в безубыток + 1 пункт;

pf\_level\_2 - уровень профита (пунктов), при котором стоплосс переносим с +1 на расстояние pf\_level\_1 пунктов от курса открытия;

pf\_level\_3 - уровень профита (пунктов), при котором стоплосс переносим с pf\_level\_1 на pf\_level\_2 пунктов от курса открытия (на этом действия функции заканчиваются);

ls\_level\_1 - расстояние от курса открытия в сторону "лосса", на котором будет установлен стоплосс при достижении профитом позиции +1 (т.е. при +1 стоплосс будет поджат на ls\_level\_1);

ls\_level\_2 - расстояние от курса открытия в "лоссе", на котором будет установлен стоплосс при условии, что курс сначала опускался ниже ls\_level\_1, а потом поднялся выше (т.е. имели лосс, но он начал уменьшаться - не допустим его повторного увеличения);

ls\_level\_3 - расстояние от курса открытия "минусе", на котором будет установлен стоплосс при условии, что курс снижался ниже ls\_level\_2, а потом поднялся выше;

trlinloss - указатель того, следует ли передвигать стоплосс на "лоссовом" участке, т.е. в интервале между начальным стоплоссом и курсом открытия (true - тралим, false - трейлинг начинается только при условии, что новый стоплосс "лучше" курса открытия, "в профите").

Автором идеи предлагалось на EURUSD использовать следующие значения: "5, 10, 25", т.е. при +5 (т.е. "отбили" спрэд и ещё 5 пунктов прошли в профит) стоплосс переносим на +1 (при ручной торговле предлагается "держать" его "в голове", дабы не искушать лишний раз диллера), при +10 п. стоплосс переносится на +5, при +25 п. профита стоплосс - на +10 и курс "отпускаем" (можно поджимать другими способами). Аналогичная часть функции для лоссового интервала автором не описана, была придумана читателями его рассылки в ходе обсуждения. Существуют сведения, что автор (весьма успешно) применяет данную методику при торговле на счете своего инвестиционного проекта, что, вероятно, является одним из факторов очень высокого % профитов в его торговле. В то же время следует отметить, что данный алгоритм трейлинга довольно специфичен и может подойти не всем.

8. Трейлинг по ценовому каналу (добавлен по совету rebus).

void TrailingByPriceChannel(int iTicket,int iBars\_n,int iIndent)

Параметры:

iTicket - уникальный порядковый номер ордера (выбранный перед вызовом функции с помощью OrderSelect());

iBars\_n - период канала (кол-во баров, среди котрых ищем наибольший хай и наименьший лоу - верхнюю и нижнюю границы канала соответственно);

iIndent - отступ (пунктов), с которым устанавливаем стоплосс от границы канала.

В принципе, по конечному результату трейлинг по ценовому каналу практически совпадает с трейлингом по теням свечей (см. выше), а также в некоторой степени напоминает трейлинг по фракталам, хоть и отличается по "концепции" и "целесообразности".

9. Трейлинг по скользящему среднему.

void TrailingByMA(int iTicket,int iTmFrme,int iMAPeriod,int iMAShift,

int MAMethod,int iApplPrice,int iShift,int iIndent)

Идея, в принципе, "в полушаге" от трейлинга по ценовому каналу. Функции передаётся тикет позиции, необходимый таймфрейм, отступ от средней (пунтов), на котором устанавливается стоплосс и собственно параметры среднего (аналогично стандартным в iMA()). Учитывая природу мувингов, несложно догадаться, что данный алгоритм может давать весьма неплохое поджатие при тренде, во флэте, однако, "путаясь" с курсом и часто приводя к преждевременному закрытию. Впрочем, можно попытаться менять параметры трала (в частности, период и метод усреднения) адекватно рыночной ситуации (учитывая волатильность, например).

Параметры:

iTicket - уникальный порядковый номер ордера (выбранный перед вызовом функции с помощью OrderSelect());

iTmFrme - период чарта, на котором будет расчитываться мувинг; допустимые варианты ввода: 1 (M1), 5 (M5), 15 (M15), 30 (M30), 60 (H1), 240 (H4), 1440 (D), 10080 (W), 43200 (MN);

iMAPeriod - период усреднения для вычисления скользящего среднего;

iMAShift - сдвиг индикатора относительно ценового графика;

iMAMethod - метод усреднения; допустимые варианты ввода: 0 (MODE\_SMA), 1 (MODE\_EMA), 2 (MODE\_SMMA), 3 (MODE\_LWMA);

iApplPrice - используемая цена; варианты ввода: 0 (PRICE\_CLOSE), 1 (PRICE\_OPEN), 2 (PRICE\_HIGH), 3 (PRICE\_LOW), 4 (PRICE\_MEDIAN), 5 (PRICE\_TYPICAL), 6 (PRICE\_WEIGHTED);

iShift - сдвиг относительно текущего бара на указанное количество периодов назад;

iIndent - отступ (пунктов) от значения среднего при размещении стоплосса.

10. Трейлинг "половинящий" (Fifty-fifty trailing).

void TrailingFiftyFifty(int iTicket,int iTmFrme,double dCoeff,

bool bTrlinloss)

Идея следующая: по закрытию очередного бара уменьшаем расстояние между стоплоссом и текущим курсом в dCoeff раз (исходно я взял 0,5, т.е. наполовину, откуда и название). Т.е., допустим, открыта покупка со стоплоссом в 40 пунктов. По закрытию бара, на котором был совершен вход в рынок, Bid оказался выше курса открытия на 42 пункта. Если мы выбрали вариант трала только "в профите" (bTrlinloss==true), то берем расстояние от курса открытия до текущего курса - 42 п., множим его на dCoeff (например, 0.5), и получив 21 п., переносим стоплосс в +21. Пусть при закрытии следующего бара профит составил +71 п. Тогда разница между текущим стоплоссом и курсом: 71-21=50, половина от найденного значения - 50\*0.5=25, и новый стоплосс должен быть установлен на 25 п. выше предыдущего (21+25=46 п. от курса открытия).

При описанном вариант трейлинга, "в профите" (bTrlinloss==true), стоплосс переносится лишь при условии, что новый стоплосс будет "лучше" курса открытия. Если же установить bTrlinloss равным false, то трал будет осуществляться и на "лоссовом" участке (т.е. интервале между курсом открытия и стоплоссом, который, кстати, должен быть обязательно определён (не равен 0), чтобы данный элемент сработал). Т.е., если взять вышеописанный вариант, то по закрытии первого бара стоплосс будет перемещен на 0.5 дистанции не между открытием и текущим курсом, а между стоплоссом и текущим курсом (при стоплоссе в 40 п. и профите в 42 п. это расстояние составит (40+42)/2 = 82/2 = 41 п., стоплосс будет установлен в +1 п. от курса открытия. На втором баре, при профите в 71 п.: а) 71 - 1 = 70, б) 70\*0.5 = 35, в) 1 + 35 = 36 п. Как видим, такой вариант будет стартовать с большей "дистанции" и несколько отставать от первого. Его основная функция - поджатие стоплосса при негативном развитии событий. Например, если по закрытию первого бара профит составил -10 п., то при bTrlinloss==true мы: а) найдем расстояние от курса до стоплосса, |-40 + (-10)| = 30 п., б) рассчитаем половину этого значения - 30\*0.5 = 15 п. и в) переместим стоплосс в сторону профита на это расстояние: -40 + 15 = -25.

При трейлинге в лоссе увеличивается количество преждевременных закрытий (причем с небольшим убытком), однако страхуемся от получения больших лоссов.

Параметры:

iTicket - уникальный порядковый номер ордера (выбранный перед вызовом функции с помощью OrderSelect());

iTmFrme - период чарта, по барам которого будет осуществляться трейлинг; допустимые варианты ввода: 1 (M1), 5 (M5), 15 (M15), 30 (M30), 60 (H1), 240 (H4), 1440 (D), 10080 (W), 43200 (MN);

dCoeff - коэффициент, определяющий то, в сколько раз будет сокращено расстояние между курсом на момент закрытия бара и текущим стоплоссом;

bTrlinloss - указатель того, следует ли осуществлять трейлинг на лоссовом участке.

11. Трейлинг "KillLoss".

void KillLoss(int iTicket,double dSpeedCoeff)

Идея такого вида сопровождения появилась у меня при разработке комбинированного вида трейлинга, в состав которого вошли несколько видов из вышеприведённых (а именно "храповой" трейлинг по Баришпольцу и трал по фракталам). Согласно задумке, он предназначен для работы только на участке лоссов (когда позиция находится в убытке). Если, например, вы поняли, что вход был неверным, лосс быстро увеличивается, курс стремительно приближается к исходному стоплоссу, может быть целесообразно ограничить вероятный убыток, перемещая стоплосс навстречу курсу, со скоростью, равной движению последнего или большей. Это, собственно, и делает функция KillLoss. После запуска она "запоминает" в глобальной переменной терминала дистанцию между стоплоссом и курсом, и в дальнейшем при приближении курса к стоплоссу на Х пунктов перемещает стоплосс на Х пунктов \* коэффициент dSpeedCoeff ему навстречу. Упомянутый коэффициент может быть константой (напр., 1 - на каждый пункт приближения курса стоплосс будет смещен на 1 пункт, или 1.5 - стоплосс будет приближаться к курсу в полтора раза быстрей движения последнего), так и значением некой переменной, которое изменяется в зависимости от состояния рынка (например, его волатильности - "есть рынок подвижен, и лосс растёт, будем урезать стоплосс быстрей"). Когда курс откатывает в сторону профита (лосс уменьшается), функция KillLoss ничего не делает. П.С. Имейте в виду, что в своей работе функция использует 2 глобальные переменные терминала - sldiff и zeticket, последняя из которых также используется в некоторых из вышеприведённых функций трейлинга. Будьте внимательны, при необходимости (напр., при совместном или попеременном применении этих функций) измените имя глобальной переменной в одной из них.

Все вышеописанные функции собраны в файле библиотеки TrailingFuncLib.mq4, прикреплённой в шапке статьи. Мне как-то удобнее использовать именно библиотеку, хоть это и несколько медленнее, однако вы можете и include'ить функции в свои программы. Прикрепляю также пример советника MyFractals с подключенными функциями. Советник, возможно, кому-нибудь тоже будет интересен.

11.08.2007. Для тех, кто торгует "полуавтоматически", контролируя часть работы вручную (открытие позиции, например, хоты бы потому, что некоторые тактики слабо алгоритмизируются), а часть - с помощью скриптов и советников на MQL4 (например, сопровождение и закрытие позиции), я написал эксперты на основе каждой из вышеописанных библиотечных функций для трейлинга. Они - в архиве TrailingExperts.zip. Так что теперь, открывшись вручную, вы можете просто запустить советник с необходимым видом трейлинга, указав ему при вызове тикет (уникальный номер позиции) и определив необходимые параметры. Причем, при необходимости, по мере изменения рыночной ситуации вы также легко можете сменить один вид трала на другой, просто запустив другого эксперта. Более подробные инструкции вы найдете в комментариях каждого из советников. Весьма надеюсь, что данные программы окажутся полезными в вашей работе.

Я занялся изучением языка MQL4 относительно недавно (в конце марта 2007). Поэтому с признательностью приму любые конструкутивные замечания, советы, предложения, идеи и прочее относительно представленных выше функций. Надеюсь, они будут для Вас полезными.

С уважением,

Юрий Дзюбан.

51 комментарий: 1 2 3 4 5 6

Авторизуйтесь или зарегистрируйтесь, чтобы добавить новый комментарий

Уважаемый Юрий. К сожалению в кодах функциях присутствуют незначительные баги, которые приводят к ошибкам в работе трала. Много сообщений об ошибках выводится в лог стандартные error 1 error 130. пустячек а не приятно. Не у всех получится быстро разобраться что к чему. Но все равно спасибо за проделаную работу. Я сэкономил много времени.

27.11.2008 16:45 SHOOTER777

komposter писал(а):

И надо бы предусмотреть одновременную работу нескольких экспертов.

Слово за слово, и написал =)

Функция KillLoss выглядит так:

void KillLoss(int iTicket,double dSpeedCoeff)

{

// проверяем переданные значения

if ((iTicket==0) || (!OrderSelect(iTicket,SELECT\_BY\_TICKET)) || (dSpeedCoeff<0.1))

{

Print("Трейлинг функцией KillLoss() невозможен из-за некорректности значений переданных ей аргументов.");

return(0);

}

double dStopPriceDiff; // расстояние (пунктов) между курсом и стоплоссом

double dToMove; // кол-во пунктов, на которое следует переместить стоплосс

// текущий курс

double dBid = MarketInfo(OrderSymbol(),MODE\_BID);

double dAsk = MarketInfo(OrderSymbol(),MODE\_ASK);

// текущее расстояние между курсом и стоплоссом

if (OrderType()==OP\_BUY) dStopPriceDiff = dBid - OrderStopLoss();

if (OrderType()==OP\_SELL) dStopPriceDiff = (OrderStopLoss() + MarketInfo(OrderSymbol(),MODE\_SPREAD)\*MarketInfo(OrderSymbol(),MODE\_POINT)) - dAsk;

string GL\_name = StringConcatenate( "KillLossTrailing\_", iTicket );

if ( IsTesting() ) GL\_name = StringConcatenate( "\_tester\_", GL\_name );

if (!GlobalVariableCheck(GL\_name))

{

GlobalVariableSet(GL\_name,dStopPriceDiff);

}

else

{

// итак, у нас есть коэффициент ускорения изменения курса

// на каждый пункт, который проходит курс в сторону лосса,

// мы должны переместить стоплосс ему на встречу на dSpeedCoeff раз пунктов

// (например, если лосс увеличился на 3 пункта за тик, dSpeedCoeff = 1.5, то

// стоплосс подтягиваем на 3 х 1.5 = 4.5, округляем - 5 п. Если подтянуть не

// удаётся (слишком близко), ничего не делаем.

// кол-во пунктов, на которое приблизился курс к стоплоссу с момента предыдущей проверки (тика, по идее)

dToMove = (GlobalVariableGet(GL\_name) - dStopPriceDiff) / MarketInfo(OrderSymbol(),MODE\_POINT);

// записываем новое значение, но только если оно уменьшилось

if (dStopPriceDiff<GlobalVariableGet(GL\_name))

GlobalVariableSet(GL\_name,dStopPriceDiff);

// дальше действия на случай, если расстояние уменьшилось (т.е. курс приблизился к стоплоссу, убыток растет)

if (dToMove>0)

{

// стоплосс, соответственно, нужно также передвинуть на такое же расстояние, но с учетом коэфф. ускорения

dToMove = MathRound(dToMove \* dSpeedCoeff) \* MarketInfo(OrderSymbol(),MODE\_POINT);

// теперь проверим, можем ли мы подтянуть стоплосс на такое расстояние

if (OrderType()==OP\_BUY)

{

if (dBid - (OrderStopLoss() + dToMove)>MarketInfo(OrderSymbol(),MODE\_STOPLEVEL)\* MarketInfo(OrderSymbol(),MODE\_POINT))

OrderModify(iTicket,OrderOpenPrice(),OrderStopLoss() + dToMove,OrderTakeProfit(),OrderExpiration());

}

if (OrderType()==OP\_SELL)

{

if ((OrderStopLoss() - dToMove) - dAsk>MarketInfo(OrderSymbol(),MODE\_STOPLEVEL) \* MarketInfo(OrderSymbol(),MODE\_POINT))

OrderModify(iTicket,OrderOpenPrice(),OrderStopLoss() - dToMove,OrderTakeProfit(),OrderExpiration());

}

}

}

}

После ее использования в тестере необходимо удалить ненужные переменные (иначе при повторном тесте трейлинг будет работать неправильно):

int deinit()

{

GlobalVariablesDeleteAll( "\_tester\_" );

return(0);

}

21.11.2008 02:14 komposter

komposter писал(а):

Это конструктивное предложение ;)

И надо бы предусмотреть одновременную работу нескольких экспертов.

Я бы вообще сделал имя переменной, хранящей расстояние до СЛ = тикету ордера. Тогда для каждого ордера была бы своя переменная и не было бы конфликтов.

21.11.2008 01:29 komposter

Я занялся изучением языка MQL4 относительно недавно (в конце марта 2007). Поэтому с признательностью приму любые конструктивные замечания, советы, предложения, идеи и прочее относительно представленных выше функций. Надеюсь, они будут для Вас полезными.

При работе с Глобальными переменными терминала хорошим тоном является использование разных имен в случае тестирования и работы на реале. Иначе тестируемый советник может нарушить алгоритм работы советника, работающего на реале.

Это конструктивное предложение ;)

21.11.2008 01:16 komposter

YuriF писал(а):

Я не программист, трейдер и наверное тут я не в "своей компании". Мне многое не понятно, какой трал в описании, какому соответствует в библиотеке? Зачем их столько много? Почему они не работают одновременно? Какова результативность их применения? и т.д.

Друзья, а нет ли у Вас, для начала, ОДНОГО максимально простого трала, который при достижении заданного профита устанавливает стоплосс в безубыток? Причём надо что бы одновременно работало несколко таких тралов на одной валютной паре.

Здравствуйте, YuriF.

Ну почему же вы так сразу "противопоставляетесь"? Согласен, иногда можно наблюдать примеры того, когда "кодится скорее ради кода, чем собственно торговли", однако все, что я до сих пор писал, так или иначе было нужно для решения каких-то торговых задач. Не скажу, что использую в своей торговле все представленные трейлинговые функции, однако некоторые - регулярно (трейлинг "по теням" и "по фракталам"). Относительно ваших вопросов:

а) названия тралов в описании и в библиотеке - в описании каждой функции приведено ее конкретное названиие, напр., трейлингу, который я назвал "стандартным-ступенчастым" (из-за особенностей алгоритма, соответствует функция

void TrailingStairs(int ticket,int trldistance,int trlstep);

Я не назвал ее TrejlingStandartnyjStupenchastyj, но указываю конкретное название, мне казалось, что разобраться не так сложно;

б) "Зачем их (функций трейлинга) столько много?" и "Почему они не работают одновременно?" Функций много, чтобы трейдер мог выбрать ту, которая, на его взгляд, наилучшим образом подходит для конкретной торговой системы (советника). Относительно второй части вопроса: YuriF, а как вы представляете себе одновременный трейлинг одной и той же позиции на расстоянии в 30 п. и 15 п.?

в) результативность применения. так уж повелось, что разные торговые системы открывают позиции не в одних и тех же местах, соответственно, тот вариант трейлинга, который хорош для одной, скорее всего, не подойдет для другой. У меня есть определенные сравнительные результаты разных видов трала для своих систем, но, боюсь, что они мало чем будут полезны вам или кому-то другому;

г) насчет переноса в БУ при определенном профите - пожалуйста, попробуйте;

д) "Причём надо чтобы одновременно работало несколко таких тралов на одной валютной паре". Это как? Одна позиция - один трейлинг (одновременно). Сколько позиций "на одной валютной паре" откроете, столько и тралить можно.

YuriF, желаю вам успеха в торговле и самообразовании! С уважением.

03.06.2008 23:28 I\_D

Здравствуйте, Ultramarin.

Извините, что отвечаю с таким запозданием (скорее всего, вопрос либо решен, либо неактуален), но все же - в советнике (в т.ч. приведеннном) функцию трейлинга следует вызывать где-нибудь из стандартной функции start(), напр.:

...

start()

{

...

...

// другие функции, вызываемые на каждом тике курса

...

// ищем открытые сделки и вызываем соответствующую функцию трейлинга, передав ей тикет открытой позиции

for (i=0;i<OrdersTotal();i++)

{

if (!OrderSelect(i,SELECT\_BY\_POS,MODE\_TRADES)) break;

if (OrderMagicNumber()!=magic || OrderSymbol()!=Symbol()) continue;

if ((OrderType()==OP\_BUY) || (OrderType()==OP\_SELL))

{

// !!! ПРИМЕР ВЫЗОВА ФУНКЦИЙ ТРЕЙЛИНГА !!!

// среди возможных вариантов мы, допустим, выбрали трейлинг по фракталам. Трейлингуем по

// 5-барным фракталам на дневках, с отступом от экстремума в 3 п., в зоне лоссов не тралим

TrailingByFractals(OrderTicket(),1440,5,3,false);

// (как видим, достаточно предварительно выбрать ордер OrderSelect() и вызвать функцию,

// передав ей тикет позиции и определив необходимые параметры).

// При желании Вы можете закоментировать данный вид трейлинга и подключить любой другой

// или даже "сконструировать" из них более или менее сложную конструкцию.

//TrailingByShadows(OrderTicket(),60,iTrlBars,3,true);

//TrailingRatchetB(OrderTicket(),10,25,30,10,25,30,false);

//TrailingStairs(OrderTicket(),50,10);

//TrailingByATR(OrderTicket(),1440,5,1,20,1,1,false);

// и т.д.

}

}

return(0);

}

03.06.2008 23:03 I\_D

Я не программист, трейдер и наверное тут я не в "своей компании". Мне многое не понятно, какой трал в описании, какому соответствует в библиотеке? Зачем их столько много? Почему они не работают одновременно? Какова результативность их применения? и т.д.

Друзья, а нет ли у Вас, для начала, ОДНОГО максимально простого трала, который при достижении заданного профита устанавливает стоплосс в безубыток? Причём надо что бы одновременно работало несколко таких тралов на одной валютной паре.

03.06.2008 19:40 YuriF

Огромная просьба ткнуть носом куда именно нужно вставить код трала ???

//+------------------------------------------------------------------+

//| Gray.mq4 |

//| Copyright © 2007, MetaQuotes Software Corp. |

//| http://www.metaquotes.net |

//+------------------------------------------------------------------+

#property copyright "Copyright © 2007, MetaQuotes Software Corp."

#property link "http://www.metaquotes.net"

extern double lot=0.1;

//+------------------------------------------------------------------+

//| expert initialization function |

//+------------------------------------------------------------------+

int init()

{

//----

//----

return(0);

}

//+------------------------------------------------------------------+

//| expert deinitialization function |

//+------------------------------------------------------------------+

int deinit()

{

//----

//----

return(0);

}

//+------------------------------------------------------------------+

//| expert start function |

//+------------------------------------------------------------------+

int start()

{

int tiket=0;

//----

if (iVolume(NULL,0,0)==1){

Print("WATR1="+iCustom( NULL, 0, "WATR", 0, 0) );

Print("WATR2="+iCustom( NULL, 0, "WATR", 1, 0) );

if (iCustom( NULL, 0, "WATR", 0, 2)<1000 && iCustom( NULL, 0, "WATR", 1, 1)<1000) {tiket =OrderSend(Symbol(),OP\_SELL,lot,Bid,3,iCustom( NULL, 0, "WATR", 1, 0),0,NULL,16384,0,Red);}

if (iCustom( NULL, 0, "WATR", 1, 2)<1000 && iCustom( NULL, 0, "WATR", 0, 1)<1000) {tiket= OrderSend(Symbol(),OP\_BUY,lot,Ask,3, iCustom( NULL, 0, "WATR", 0, 0),0,NULL,16384,0,Green);}

CheckOrders();

//----

return(0);

}

//+------------------------------------------------------------------+

}

void CheckOrders(){

int pos=0;

int total=OrdersTotal();

for (pos=0;pos<total;pos++) {

if (OrderSelect(pos,SELECT\_BY\_POS,MODE\_TRADES)==True){

if (OrderType()==OP\_BUY && OrderStopLoss()<iCustom( NULL, 0, "WATR", 0, 1))OrderModify(OrderTicket(),OrderOpenPrice(),iCustom( NULL, 0, "WATR", 0, 1),OrderTakeProfit(),0,Green);

if (OrderType()==OP\_SELL && OrderStopLoss()>iCustom( NULL, 0, "WATR", 1, 1))OrderModify(OrderTicket(),OrderOpenPrice(),iCustom( NULL, 0, "WATR", 1, 1),OrderTakeProfit(),0,Red);

}

}

}

08.03.2008 07:49 Ultramarin

I\_D писал(а):

Idalgo писал(а):

Можно как-то модифицировать функцию KillLoss, чтобы она работала для случаев, когда одновременно открыто более одного ордера? Если, к примеру, два ордера открыто, то не тралится ничего.

Имхо, вместо модификации функции проще перед её вызовом пересчитать открытые позиции и далее, в зависимости от их числа, либо вызывать, либо нет. Например:

Юрий, Вам - большое спасибо за Ваши усилия, выложенные материалы и оперативные ответы на вопросы!

Насчет модификации функции KillLoss, я не совсем то имел в виду. Я создал советника, который открывает одновременно ордера в количестве большем, чем 1, и предполагал вставить в него в качестве опции функцию KillLoss. Замечено, что функция KillLoss корректно работает только при одном открытом в данный момент ордере, а если, скажем, одновременно открыто 2 ордера, то она выключается (не срабатывает), и стоплосс срабатывает там, где он изначально ставится.

Проще говоря, KillLoss не тралит одновременно две позиции, как, например, Udavka.

03.03.2008 11:03 Idalgo

Idalgo писал(а):

Можно как-то модифицировать функцию KillLoss, чтобы она работала для случаев, когда одновременно открыто более одного ордера? Если, к примеру, два ордера открыто, то не тралится ничего.

Имхо, вместо модификации функции проще перед её вызовом пересчитать открытые позиции и далее, в зависимости от их числа, либо вызывать, либо нет. Например:

int iOpenOrdersN; // переменная для подсчета кол-ва открытых позиций

int i; // переменная-счетчик для циклов

for (i=0;i<OrdersTotal();i++)

{

if ((OrderType()==OP\_BUY) || (OrderType()==OP\_SELL))

iOpenOrdersN++;

}

if (iOpenOrdersN==1)

// ну или

// if ((iOpenOrdersN<=1) && (iOpenOrdersN>0))

{

// трейлинг на каждом тике

for (i=0;i<OrdersTotal();i++)

{

if (!OrderSelect(i,SELECT\_BY\_POS,MODE\_TRADES)) break;

if (OrderMagicNumber()!=magic || OrderSymbol()!=Symbol()) continue;

if ((OrderType()==OP\_BUY) || (OrderType()==OP\_SELL))

{

KillLoss(OrderTicket(),1);

}

}

}

С уважением, Юрий.

02.03.2008 19:13 I\_D

MQL4: библиотека советников, индикаторов и скриптов для автоматической торговли в MetaTrader

© 2000-2009, MetaQuotes Software Corp. Все права защищены.