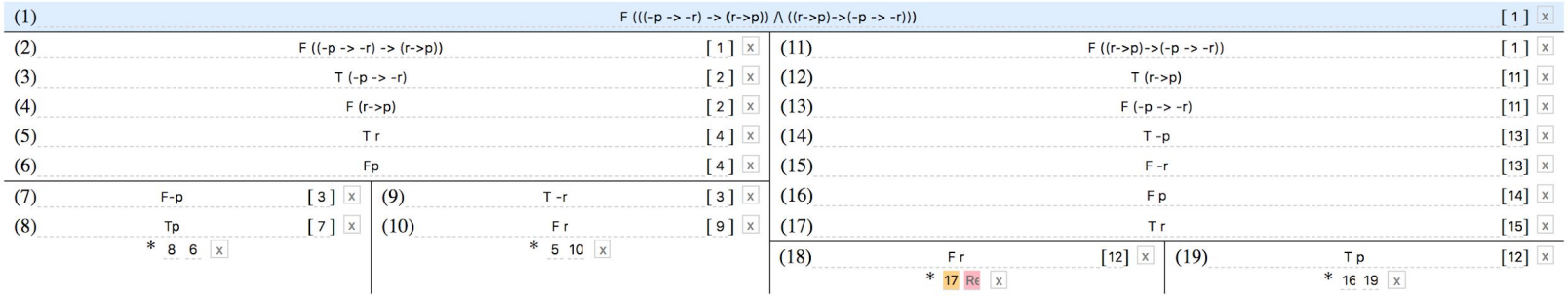
Educational tool for first order logic

Alexandra Nyitraiova

ČO ROBÍM?

-asistent pre dokazovanie tablovým kalkulom
-nedokazuje za mňa
-validuje dôkaz a vyznačuje chyby
-dôkaz vizualizuje ako stromovú štruktúru, ((-p -> -r) <-> (r->p)) je tautológia



This tableau might be proving (once correct):

⊦ (((¬p→¬r)→(r→p))∧((r→p)→(¬p→¬r)))

Problems

- (18) Second close reference is invalid.

Prettify formulas | Print | Export as JSON | Import from JSON

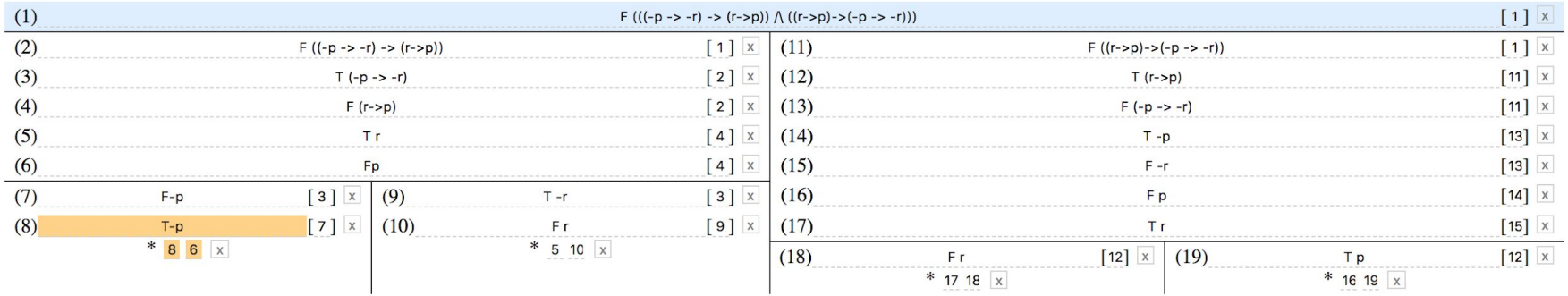
Help

Use `&`, `/\` or `∧` for conjunction, `|`, `\/` or `∨` for disjunction, `->` or `→` for implication, and `-`, `~` or `¬` for negation. Conjunction and disjunction are strictly binary. Each node of the tableau contains a signed formula, i.e. it must be prefixed by `T` or `F`.

To enter a premise / assumption (which you want to prove), make it reference itself (i.e. "(1) F ... [1]").

α-rules

T (A∧B) F (A∨B) F (A→B) T ¬A F ¬A



This tableau might be proving (once correct):

⊦ (((¬p→¬r)→(r→p))∧((r→p)→(¬p→¬r)))

Problems

- (8) Is not an α-subformula of (7).
- (8) Closing formulas are not complementary.

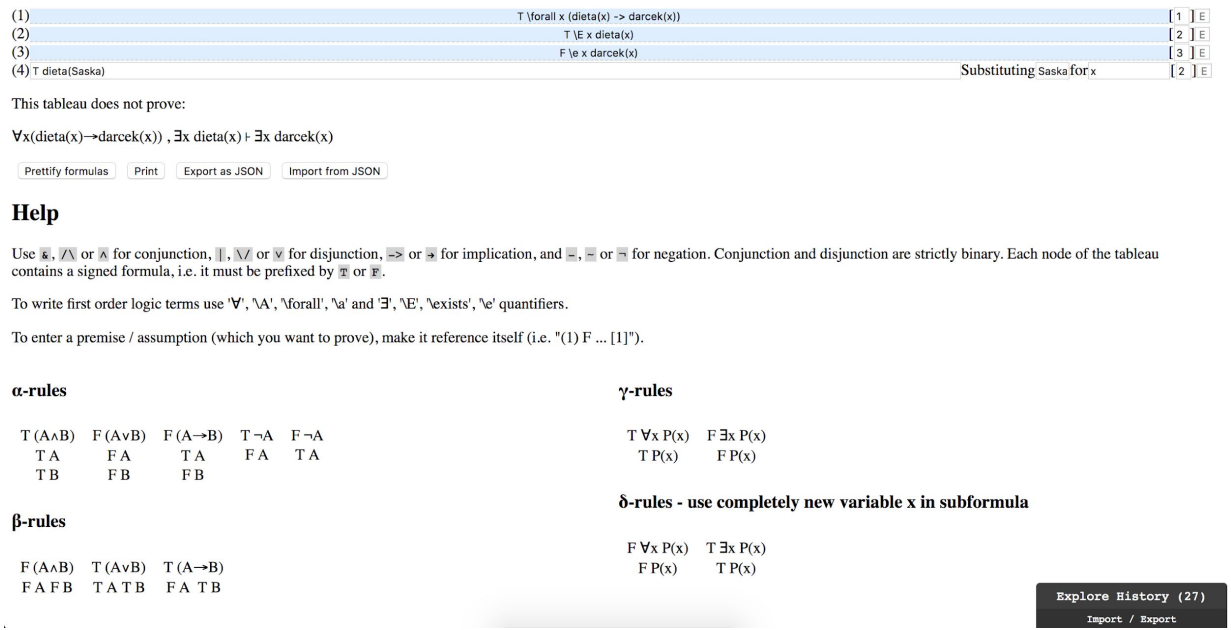
Prettify formulas | Print | Export as JSON | Import from JSON

-prettify (u)
-validácia Alfy a Bety
-export/import v JSONe
-záver
-save as pdf
-helper section
-pridavanie nodov len na konci (u)
- mazanie celého podstromu (u)

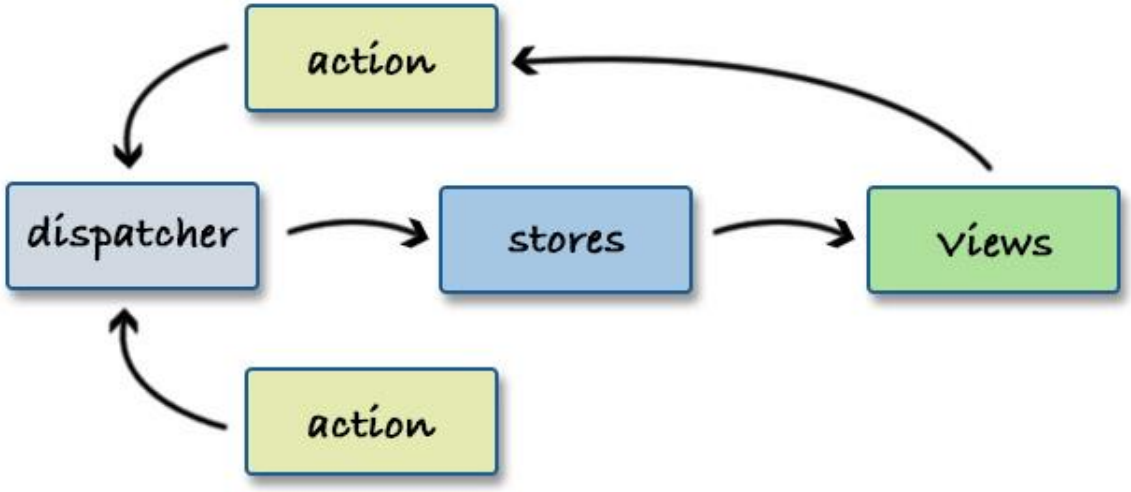
Čo je v pôvodnom?

Čo robím navyše?

-prerábka štruktúry
-renderovanie v divoch
-substitucia (transformacia a validacia)
-validacia Gammy a Delty (u)
-pridanie nodu hocikde (u)
-mazanie vybraného nodu ale aj podstromu (u)
-switch biet (u)
-zmena bety na alfu ak jeden z podstromov chýba (u)
-undo/redo



Flux flow v Elme




```
update : Msg -> Model -> ( Model, Cmd Msg )
update msg model =
    case msg of
        JsonSelected ->
            ( { model | jsonImportError = "", jsonImporting = True }, fileSelected model.jsonImportId )

        _ ->
            ( simpleUpdate msg { model | jsonImportError = "" }, Cmd.none )
```

```
20  main : Program Never Model Msg
21  main =
22      Html.program
23          { init = init
24          , update = update
25          , view = view
26          , subscriptions = subscriptions
27          }
28
29
30  type alias Model =
31      { tableau : Tableau
32      , jsonImporting : Bool
33      , jsonImportError : String
34      , jsonImportId : String
35      }
36
37
38  init : ( Model, Cmd msg )
39  init =
40      ( { tableau =
41              { node =
42                  { id = 1
43                  , value = ""
44                  , reference = { str = "1", up = Just 0 }
45                  , formula = Formula.parseSigned ""
46                  , gui = defGUI
47                  }
48              , ext = Open
49              }
50        , jsonImporting = False
51        , jsonImportError = ""
52        , jsonImportId = "importJson"
53        }
54      , Cmd.none
55      )
56
57
58  subscriptions : Model -> Sub Msg
59  subscriptions model =
60      fileContentRead JsonRead
```

```elm
simpleUpdate : Msg -> Model -> Model
simpleUpdate msg model =
    Debug.log "model"
        (case msg of
            ChangeText z new ->
                { model | tableau = z |> Zipper.setFormula new |> top }

            ExpandAlpha z ->
                { model | tableau = z |> Zipper.extendAlpha |> topRenumbered }

            ExpandBeta z ->
                { model | tableau = z |> Zipper.extendBeta |> topRenumbered }

            ExpandGamma z ->
                { model | tableau = z |> Zipper.extendGamma |> topRenumbered }

            ExpandDelta z ->
                { model | tableau = z |> Zipper.extendDelta |> topRenumbered }

            ChangeRef z new ->
                { model | tableau = z |> Zipper.setRef new |> top }

            Delete z ->
                { model | tableau = z |> Zipper.delete |> topRenumbered }

            DeleteMe z ->
                { model | tableau = z |> Zipper.deleteMe |> topRenumbered }

            MakeClosed z ->
                { model | tableau = z |> Zipper.makeClosed |> top }

            SetClosed which z ref ->
                { model | tableau = z |> Zipper.setClosed which ref |> top }

            MakeOpen z ->
                { model | tableau = z |> Zipper.makeOpen |> top }

            ChangeVariable z newVariable ->
```

```elm
view : Model -> Html Msg
view model =
    div [ class "tableau" ]
        [ viewNode (Zipper.zipper model.tableau)
        , verdict model.tableau
        , problems model.tableau
        , p [ class "actions" ]
            [ button [ onClick Prettify ] [ text "Prettify formulas" ]
            , button [ attribute "onClick" "javascript:window.print()" ] [ text "Print" ]
            , jsonExportControl model.tableau
            , jsonImportControl model
            ]
        , jsonImportError model
        , Rules.help
        ]
```

Ďakujem za pozornosť

Otázky