Automated Analog Design

Part III

A symbolic analysis approach

Implementation issues

Objectives

- Abstract
- About my last presentations
- Review related works
- Building an amplifier ontology
- Introduction of Geometric programming
- Proposal automated analog design
- Reasoning with CLIPS
- Difficulties and future goals
- Conclusions

Abstract

- Automated analog design – why?
- Some issues in automated design:
  - Logical Design level

  Knowledge retrieval in order to synthesize the logical top level of the circuit.

  - Circuit design level

  Automated module retrieval by specified Logical circuit

  - Design for IC or design for System unit

Abstract

- Possible goals
  - Automated design flow – available in most common cases
    - Prototype generation in digital design (VHDL).
    - Placement modules on die.
    - Sizing and V/I biasing of prototype circuits – optimization
    - Elements sizing in simple analog circuits such as filters, power supply, etc.
  - Automated design as an abstraction of knowledge retrieval.
  - Automated design as an abstraction of knowledge discovery.

Abstract

- Example for design automation -VHDL

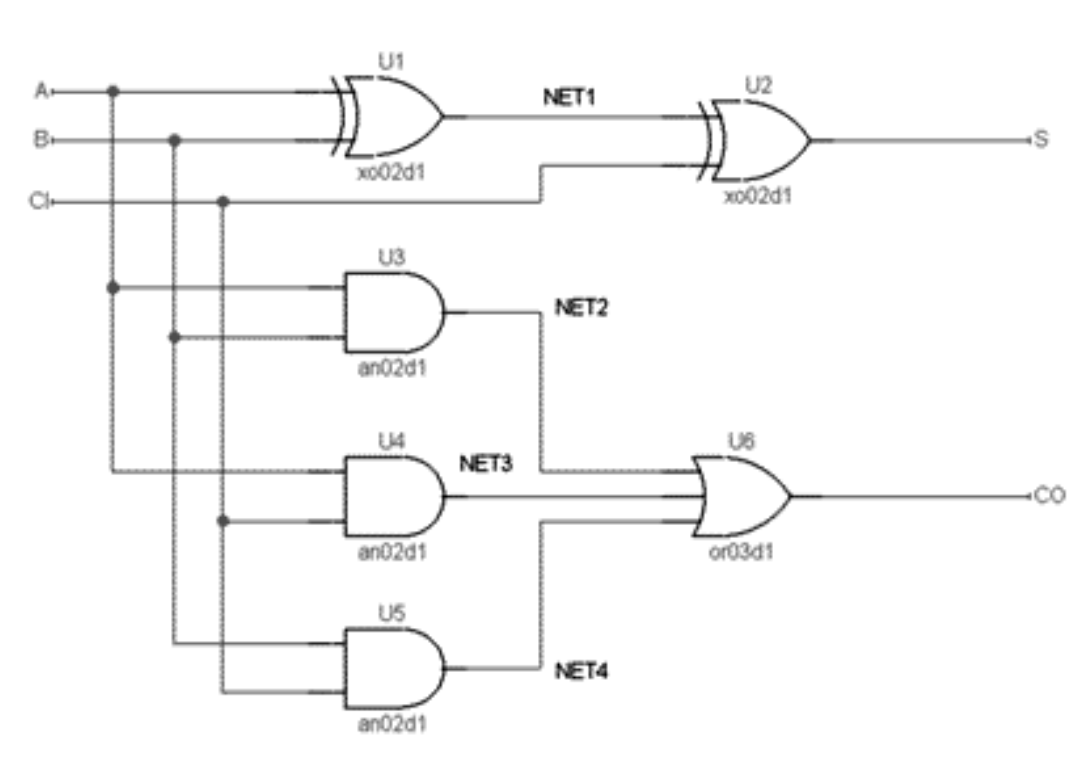
```
full_adder.java                                                    12/3/2003

ENTITY full_adder IS
  PORT (
    a  : IN BIT;
    b  : IN BIT;
    ci : IN BIT;
    s  : OUT BIT;
    co : OUT BIT
   );
END full adder;
-- ========================================

ARCHITECTURE gate level OF full adder IS
  COMPONENT an02d1 PORT (a1, a2:     IN BIT; z: OUT BIT); END COMPONENT;
  COMPONENT xo02d1 PORT (a1, a2:     IN BIT; z: OUT BIT); END COMPONENT;
  COMPONENT or03d1 PORT (a1, a2, a3: IN BIT; z: OUT BIT); END COMPONENT;
  -- Intermediate nets
  SIGNAL net1,net2, net3, net4 : BIT;
BEGIN
  -- Sum Output Bit
  U1 : xo02d1 PORT MAP (a, b, net1);
  U2 : xo02d1 PORT MAP (ci, net1, s);
  -- Carry Output Bit
  U3 : an02d1 PORT MAP (a, b, net2);
  U4 : an02d1 PORT MAP (a, ci, net3);
  U5 : an02d1 PORT MAP (b, ci, net4);
  U6 : or03d1 PORT MAP (net2, net3, net4, co);
END gate_level;
```

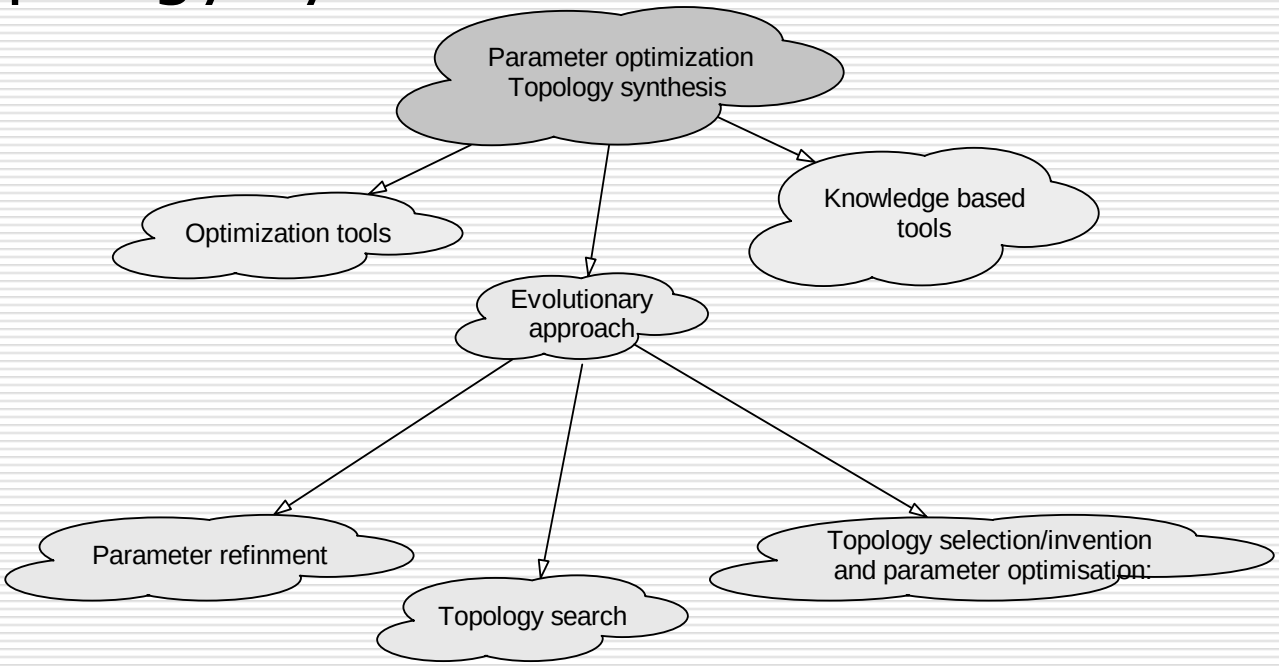
Abstract

- Example for design automation -VHDL



Review related works

- Techniques for parameter optimization and topology synthesis



Review related works

- Optimization tools:
  - **DELIGHT.SPICE,ADOPT**
  - **ECSTASY** (circuit topologies are selected from the library using heuristic design rules or the topologies are manually selected by the users and are then processed by a *symbolic simulator* or a *circuit compiler* where circuit characteristics are transformed into *cost functions)*
  - **ISAAC and OPTIMAN** – symbolic simulators
- Knowledge base for topology synthesis and parameter searches:
  - **OPASYN –** uses decision rules
  - **Ampdes, CAMPS** uses design rules for synthesis, analytical analysis for sizing

Review related works

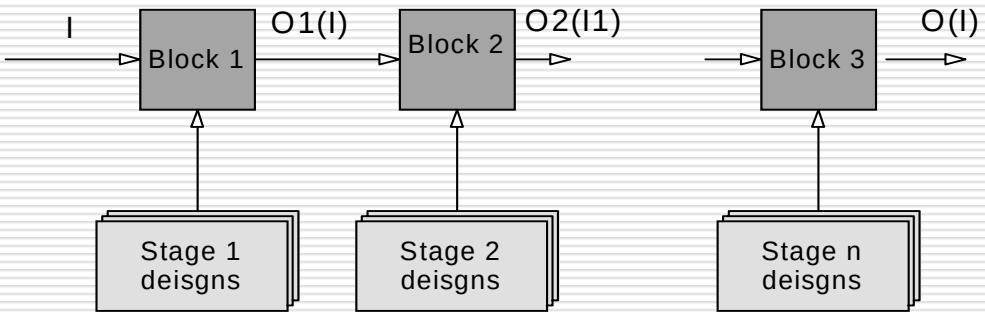
- The evolution of circuit topologies and component parameters
  - **Parameter refinement: Wo'jcikowski**: (used genetic algorithms to search for transistor length, bias voltage and current load of operational amplifiers.)
  - **Topology search: Grimbleby**: (constructed a network comprising resistors and capacitors)
  - **Topology selection/invention and parameter optimization: DARWIN**: (chose randomly topologies, then evolutionary eliminates unsuited ones. Evolutionary is finding the circuit parameters)

Symbolic computation

- Symbolic analysis
  - I’m still not sure how to perform the evaluation. One possible solution is to use some basic ontology and evolutionary approach to generate set of prototypes.
  - If we have large set of prototypes simple ontology would be followed by many computations.
  - This basically will be ontology based reasoning.
  - The important points is :
    - How would be handled non retrievable requests
    - How will be update the ontology
    - How to build 0 level ontology class where would be placed all common classes for analog circuits.

Symbolic computation

- Scenario for symbolic designer
  - The goal is to synthesize circuit with some output
  - In order to perform reasoning over the set of symbolic equitation, we need explicit parameter representation.



$$O = f(I)$$

$$O = O(p_1, p_2, \ldots, p_n)$$

Partitionning :

$$O_i = O_i(O_{i-1}(I_{i-1}))$$

Symbolic computation

- Scenario for symbolic designer
  - In evaluating stage will be calculated the overall performance with respect each of the parameters in our area of interest
  - The decision will be based on the some e fitness function:

$$F^1 = W_1 P_1^1 + W_2 P_2^1 + ...W_n P_n^1$$

$$W - weight;$$

$$P - parameter$$

$$F_{BEST} = \min(F^i)$$

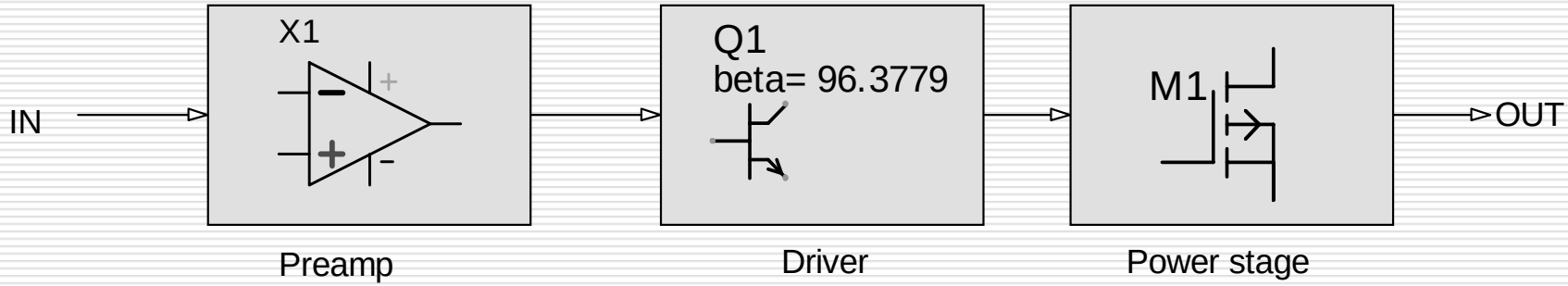
- Difficulties
  - No analytical expressions for some parameters
  - Heavy mathematical representation for others
  - I still don't know useful library for symbolic manipulation

Example: analog design process

- Consider the scenario: engineer X has to design amplifier with parameters:
  - Power 5W
  - Input 20-20khz ±3 db
  - Output 8 Ω
  - THD < 0.1
  - SN ratio >90db
  - Cost < $50*
- How many circuits are available?
- Can we consider the design as optimization process?

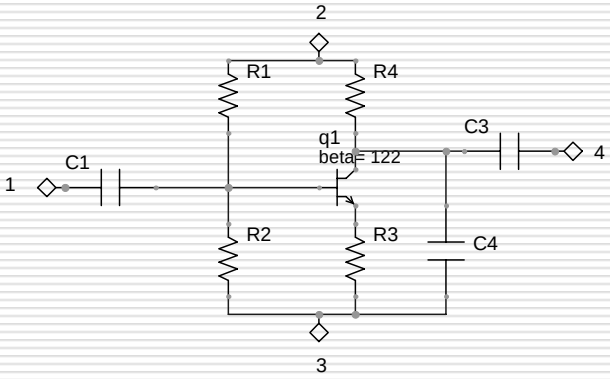
Example: analog design process

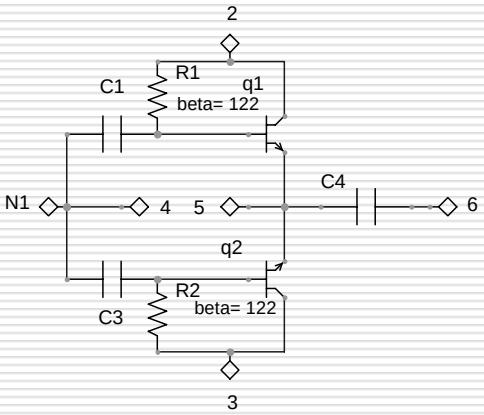
- Standard model of amplifier design
  - ≈85% of all available design
  - Allows partitioning and distinct analysis of each block

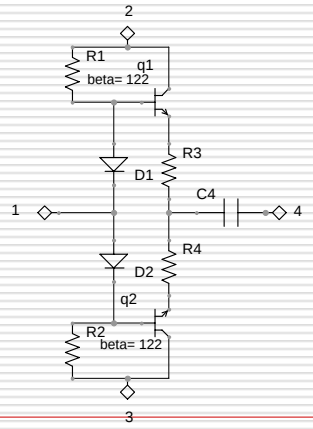


Example: analog design process

- Example: Class A, B and AB amplifier prototypes
- I explored 80 different amplifier design:
  - Design A appears in 21 (or variations)
  - Design B appears in 37 (or variations)
  - Design C appears in 66 (or variations)







Example: analog design process

- Possible solution:
  - Output power reflects to the power supply, where are certain prototypes, or “sizable design”
  - “Hi-Fi” specification reflects to circuit topology.
  - THD and SN ratio correlate with components quality
  - Output parameters reflects to topology and/or elements, which determines the overall cost.
- How many topologies we have?
  - 5 basics + 3 improved topologies
  - 1 high-end topology

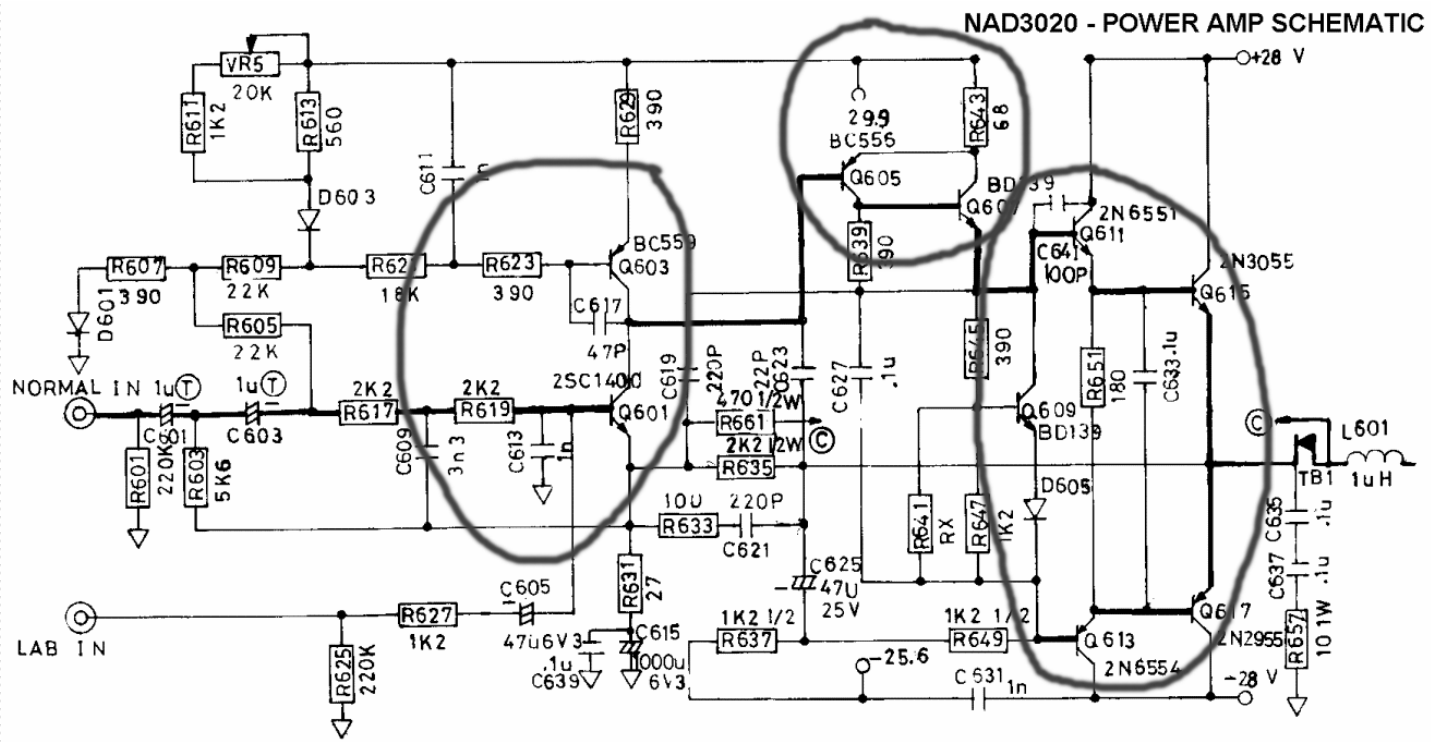
Example: analog design process

- Generated topologies:

Example: analog design process

- Conclusion:
  - The design space is not so big.
  - We can solve at least 90% of the custom specification.
  - We need as many as possible prototype circuits
  - We need as explicit specification how different topology correlates with requirement
  - We need set of elements with wide parameter variance
  - This particular design has to be solved as an optimization problem

Example: analog design process



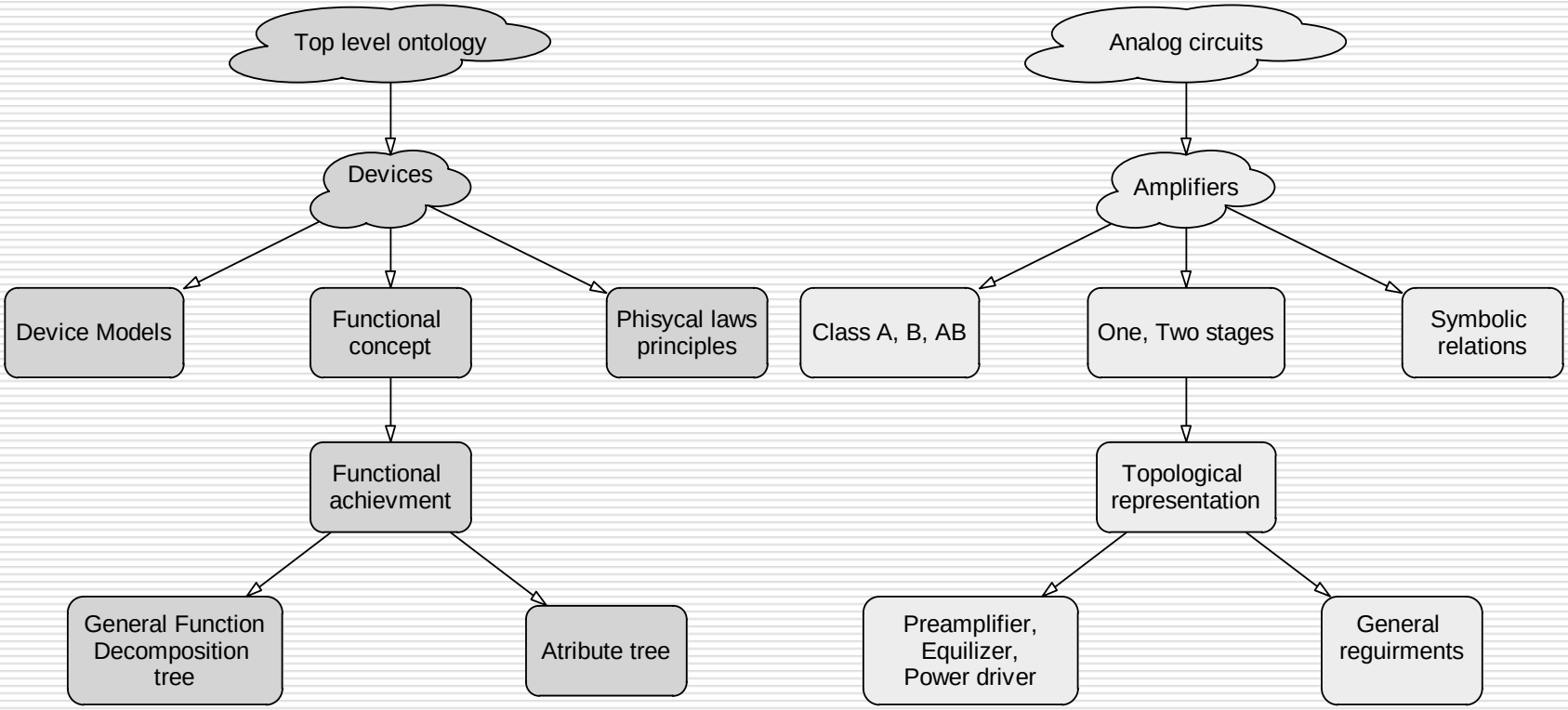
Amplifiers ontology

- Should provide knowledge about:
  - Different topologies (sizable)
  - Different output parameters (Gain, SN, Power)
  - Instantiation of topology design
  - 0 level of circuit logic (Amplifier consists: preamp, drivers, etc)
  - Input, output, internal nodes
- Other issues:
  - Has to describe all design problem aspect
  - Ability to be modified according certain rules
  - Has to determine set of the best solutions
  - Has to give enough freedom to the designer for modifying the solution

Amplifiers ontology

- Consider amplifier ontology restricted to:
  - Audio frequency
  - Power up to 50W
  - SN up to 90 db
  - One linear input
  - No more than 3 stages

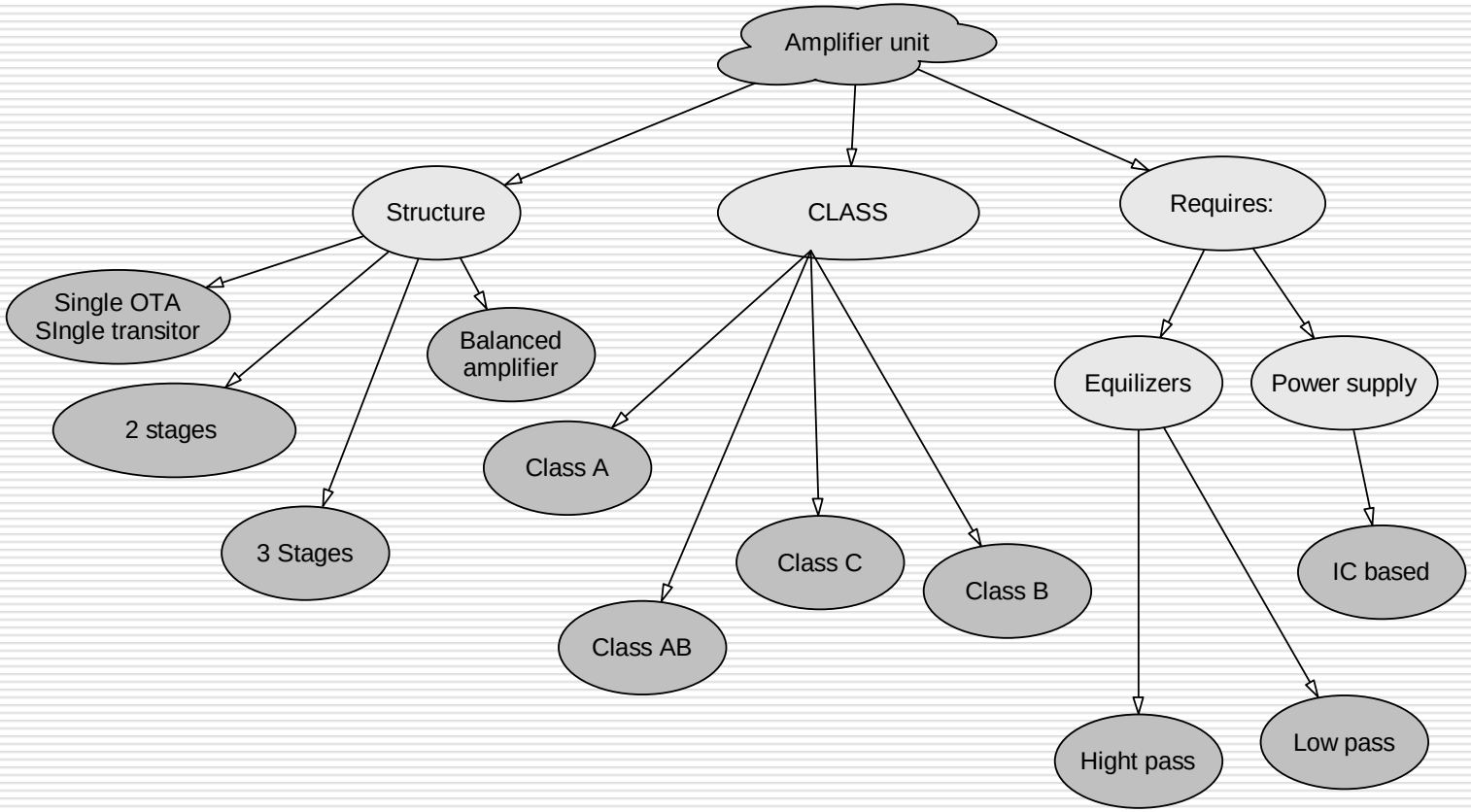
Amplifiers ontology



Amplifiers ontology

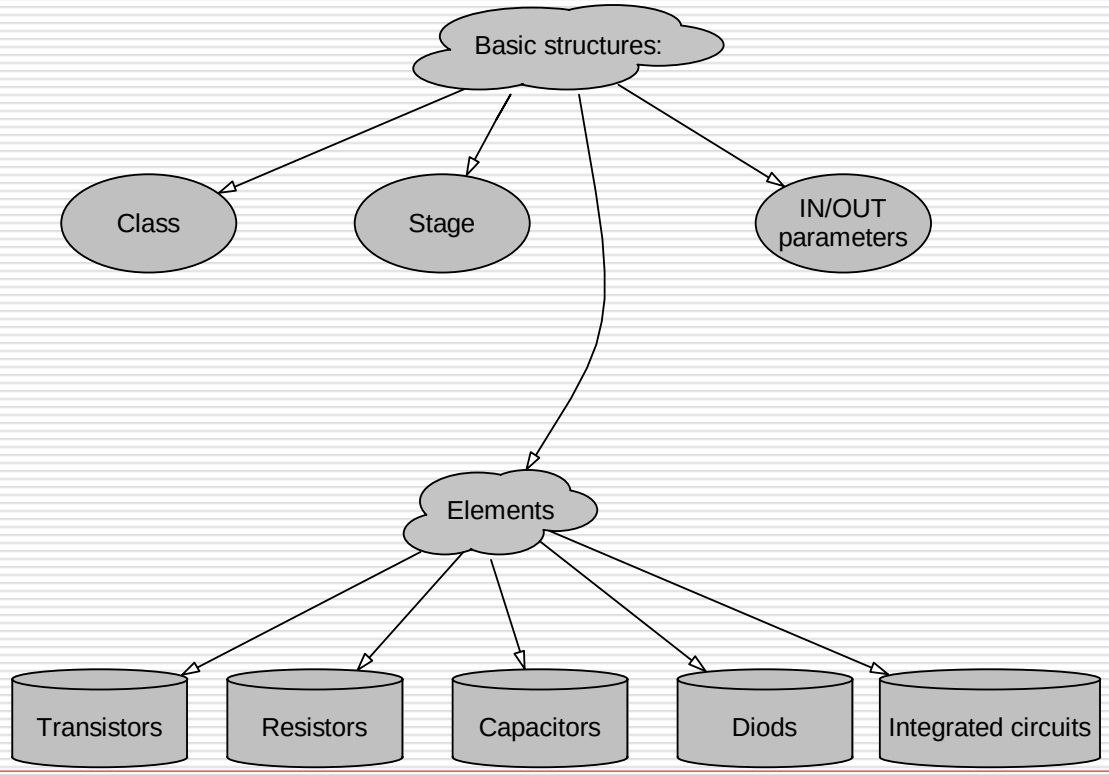
- **Device model:** describe the amplifier class, which is closely related to the basic output parameters
- **Functional concept:** describe the meta-topology level: ho many stages and what is their functional purpose
- **Physical principles:** Here should be provided symbolic equations for calculating the overall performance.
- **General function decomposition tree**: Provides information how to build these basic structures
- **General requirements:** example: Preamp Vout=±1V, Driver Vin =±1V

Amplifiers ontology

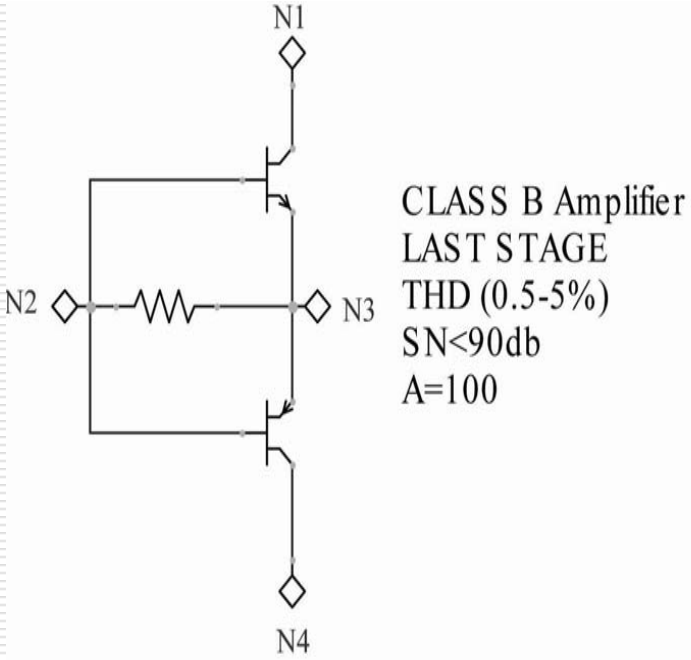


Amplifiers ontology

- **Device model:** describe the amplifier class, which is closely related to the basic output parameters



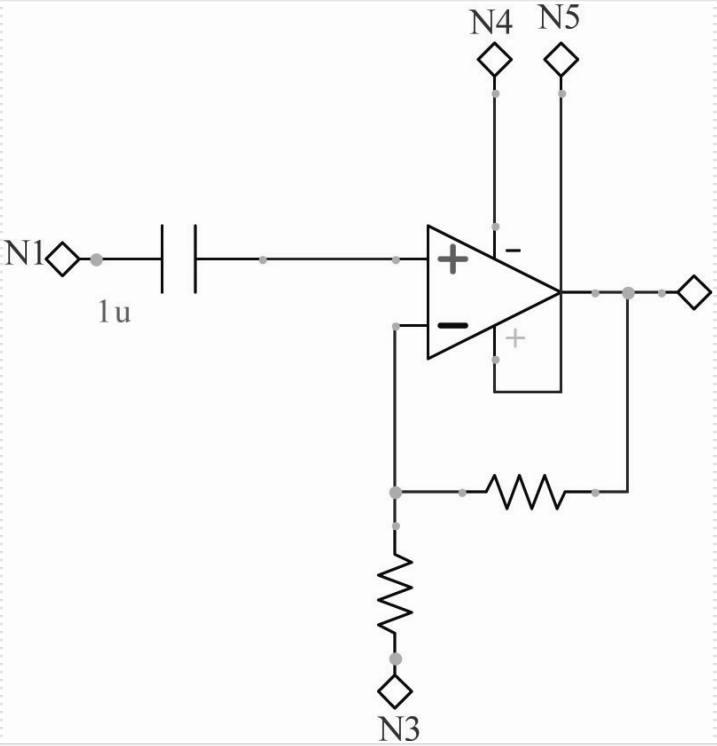
Amplifiers ontology



Example:

- Slot1 – class B
- Slot2 – Output driver
- Slot3 – THD,SN,A
- Slot4 – sets of feasible components

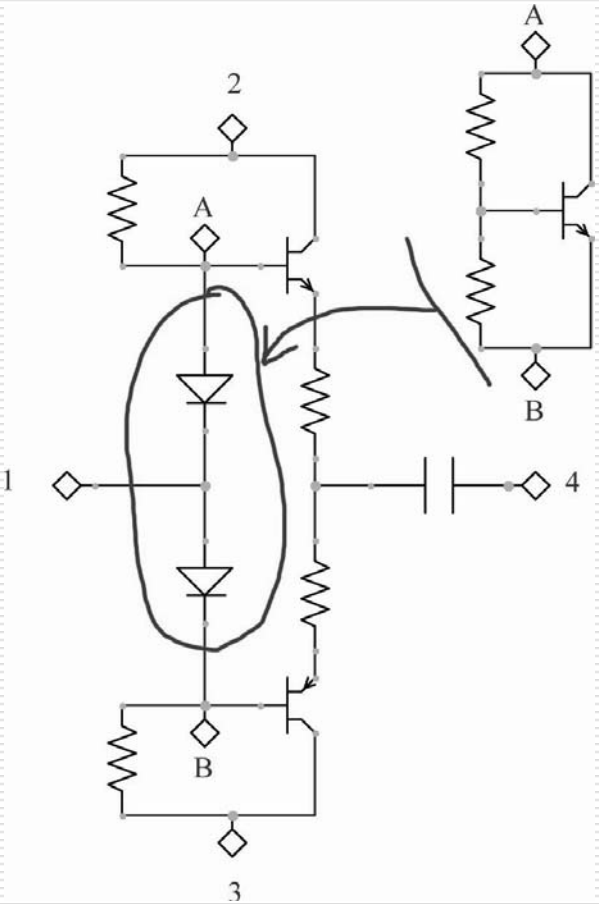
Amplifiers ontology



Example:

- Slot1 – IC
- Slot2 – Preamp
- Slot3 – THD,SN,A
- Slot4 – sets of feasible components

Amplifiers ontology

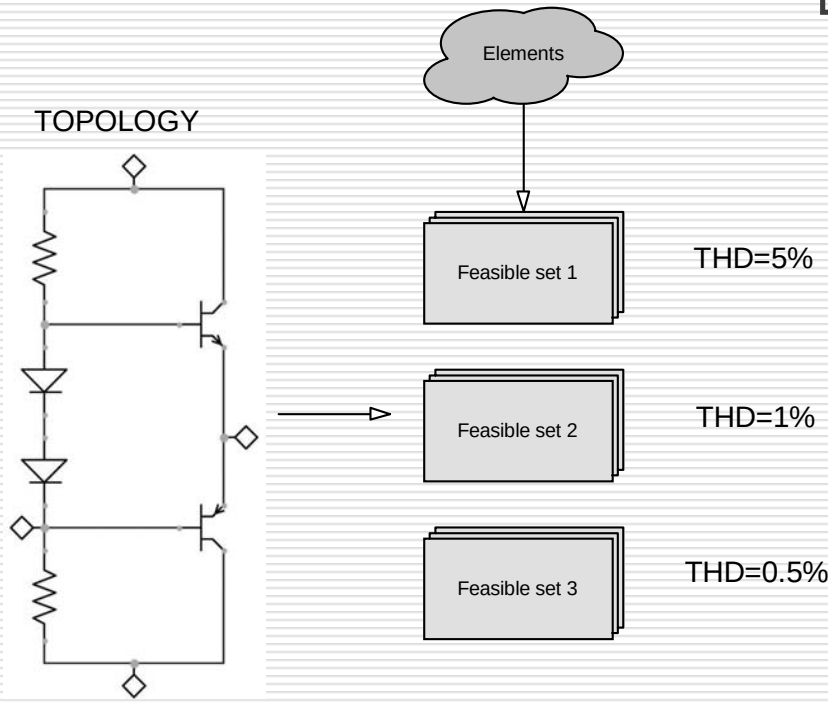


Example:

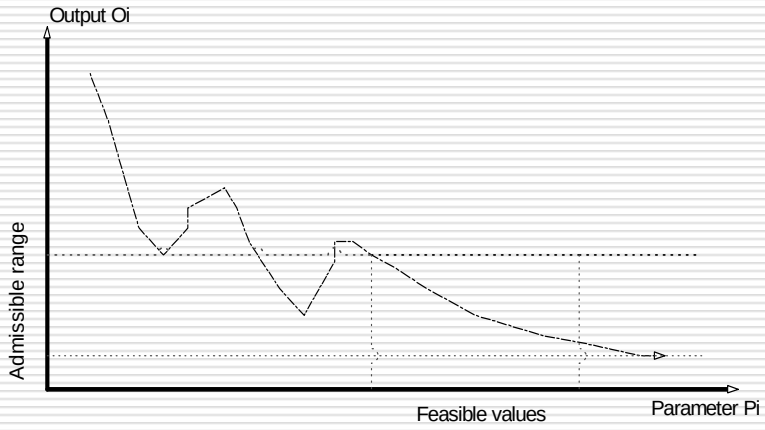
- Slot1 – class AB
- Slot2 – Output driver
- Slot3 – THD,SN,A
- Slot4 – sets of feasible components, including modified arc AB

  (reflects to better biasing the output transistors)

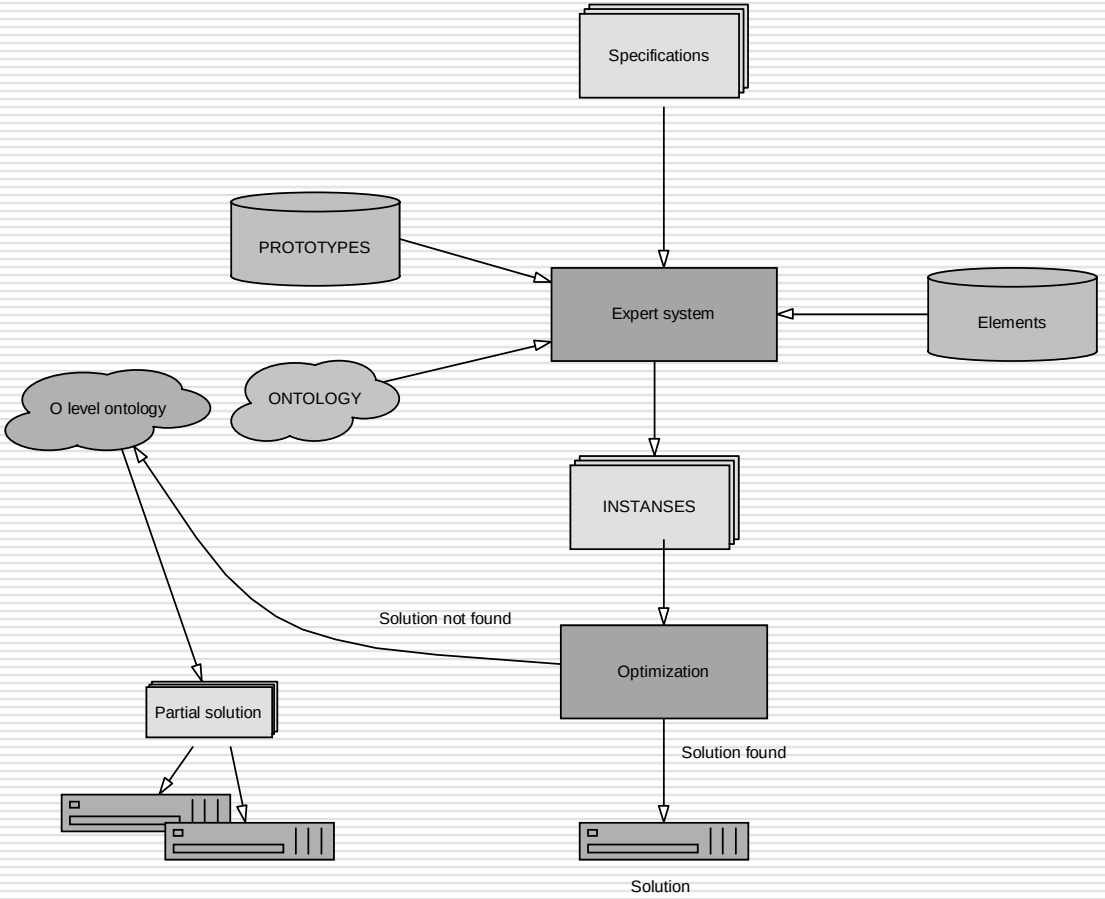
Amplifiers ontology



- In order to minimize the optimization process, each topology is mapped to feasible set of elements



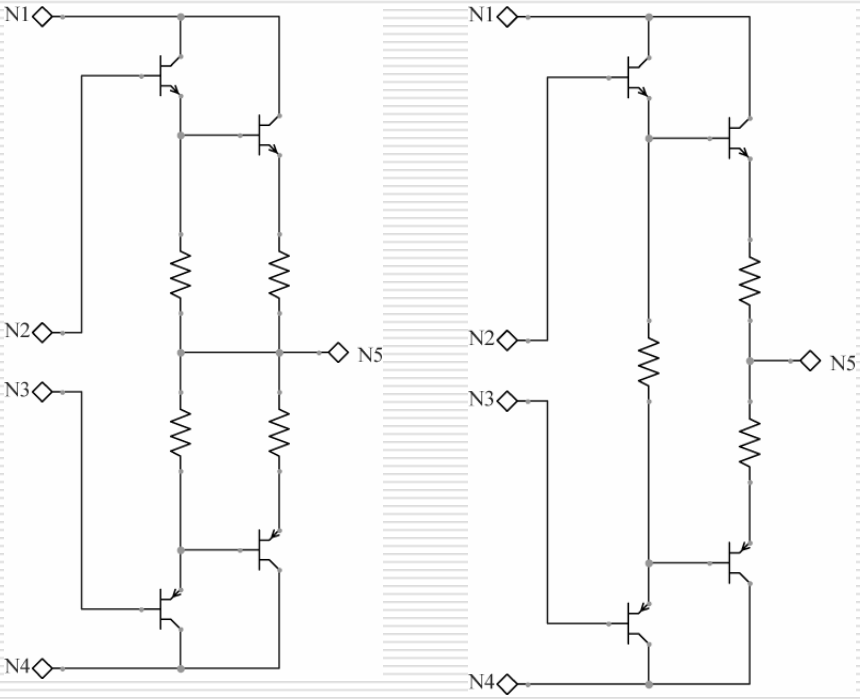
Automated amplifier design



Evolutionary approach

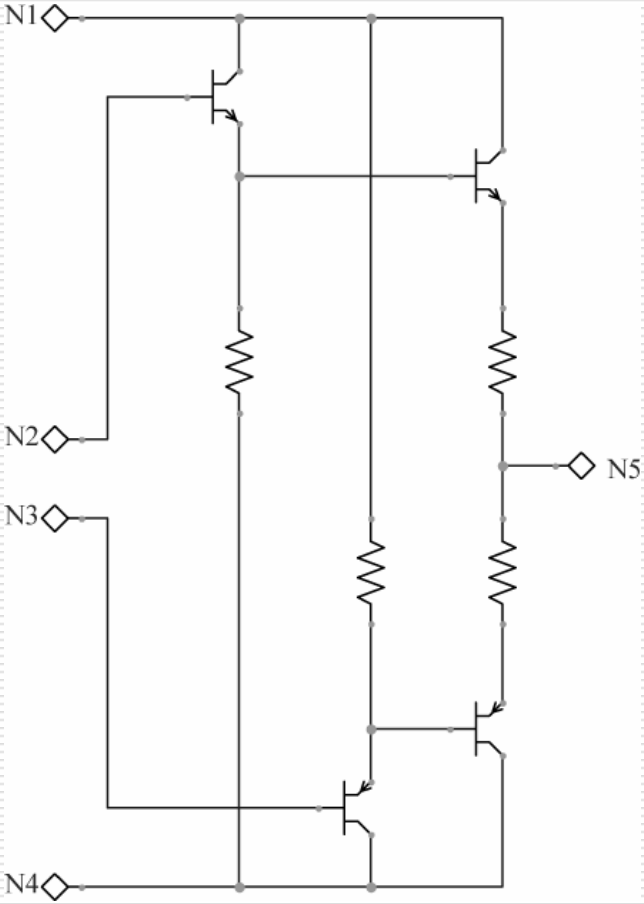
- Scenario 1 Scenario 1 – every obtained final solution we allow small variance and by using evolutionary approach, looking for better solution
- Scenario 2 – We can use evolutionary approach in the highest level of ontology

Scenario 1 - example



- Class B power stage standard implementation
- Circuit 1 has better THD parameters due to the local feedback

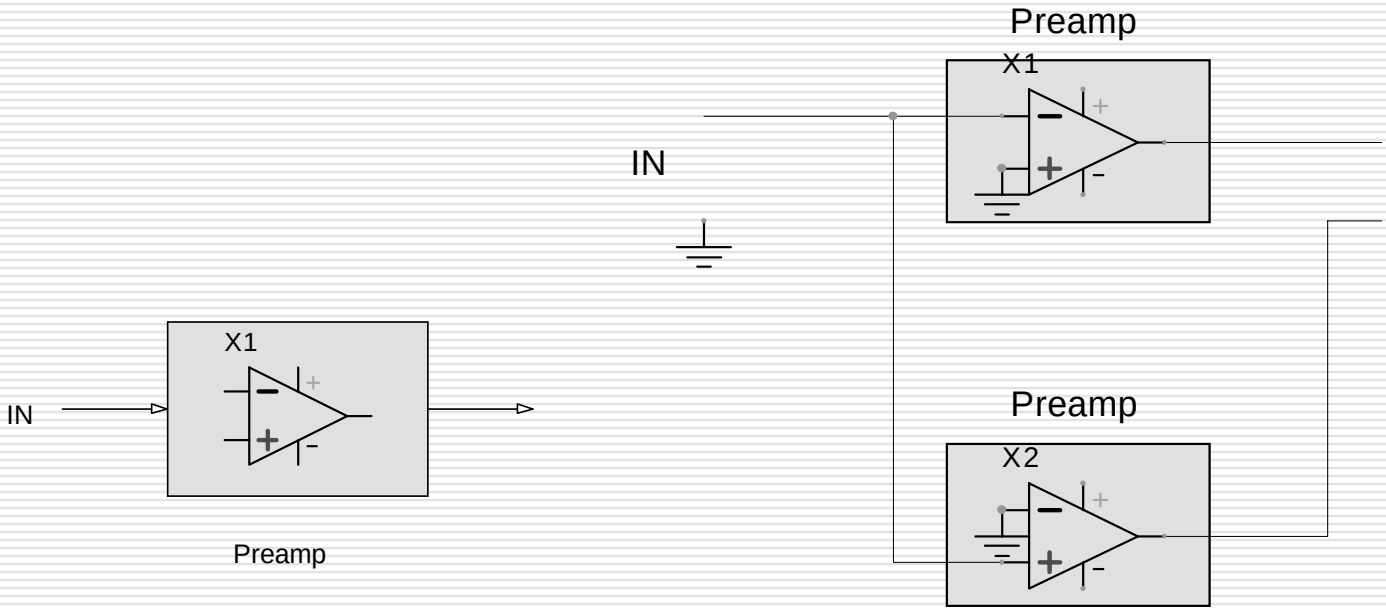
Scenario 1 - example

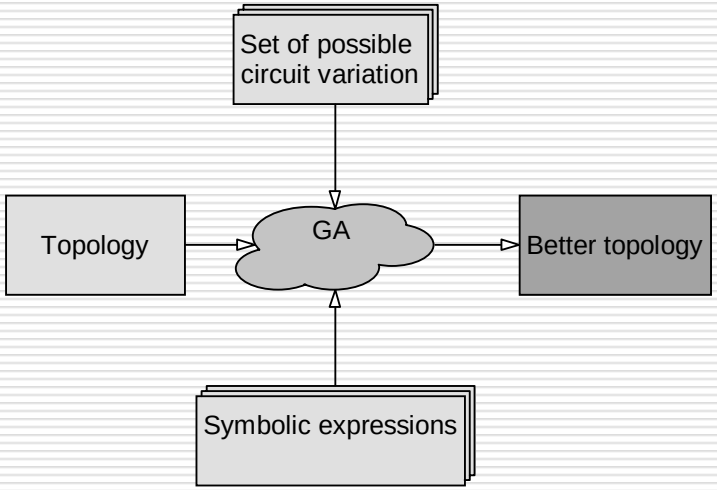


- The best choice regarding THD improvement
- Simple GA code generate this configuration with probability 1:1296
- I am looking for a good analytical expression for THD in order to calculate the fitness performance

Scenario 2 - example

- First circuit is standard preamp
- Second circuit has better SN ratio. Can be obtained with GA with probability 2:81

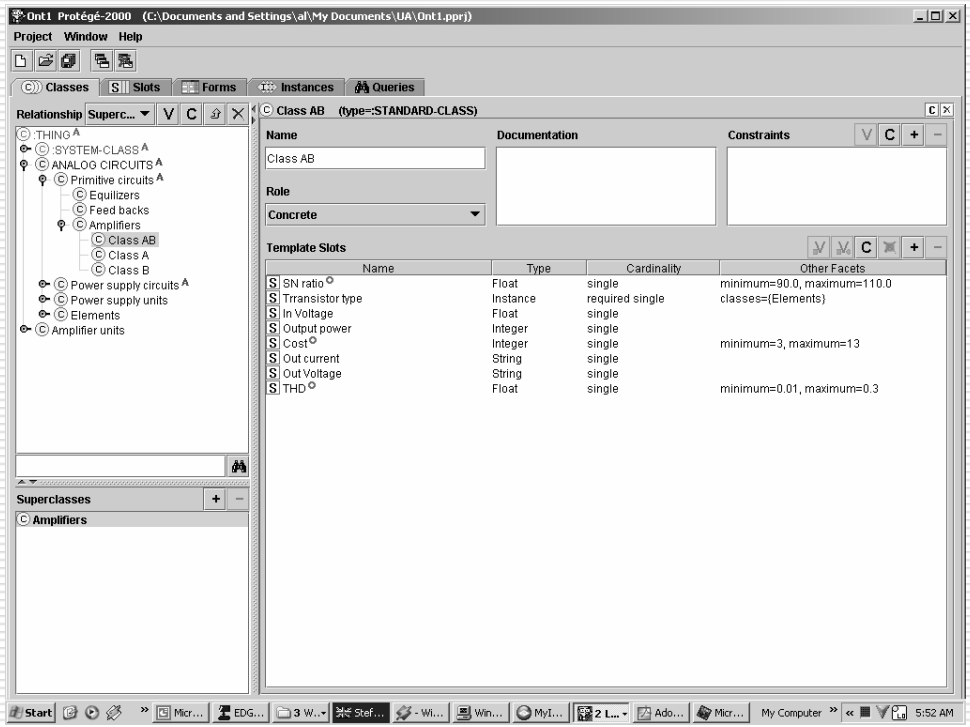




- Difficulties:
  - The circuit variations depends on the structure
  - Some variable such as THD doesn’t have explicit analytical expression
- Solutions:
  - Geometric programming
  - Empiric formulas
  - Predefined stable/floating nodes

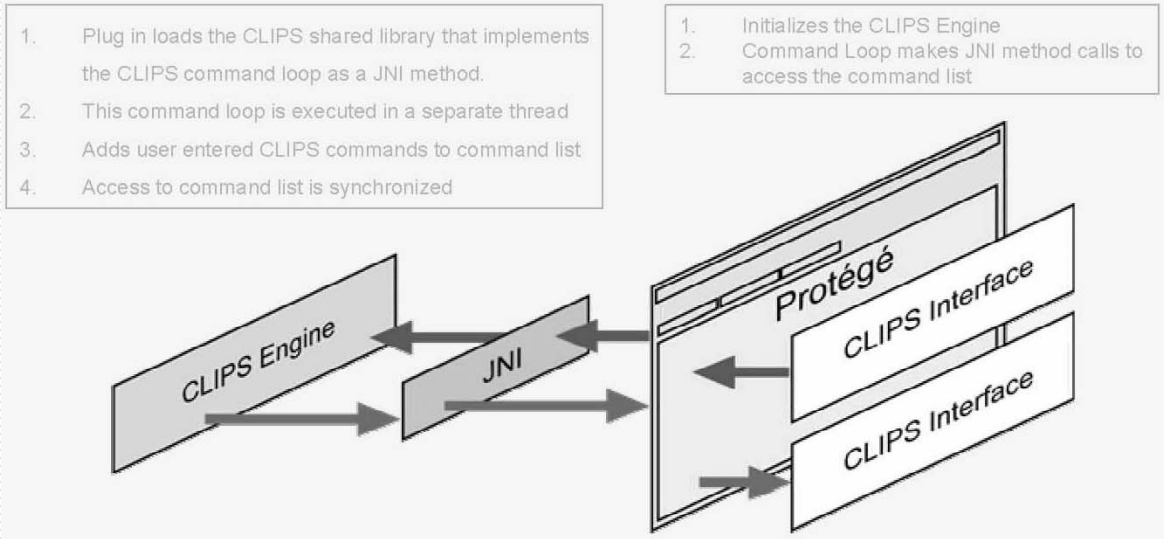
Implementation

- Protegee-2000



Implementation

- Clips
  - CLIPS is a development and delivery expert system tool providing a complete environment for the construction of rule and/or object based expert systems

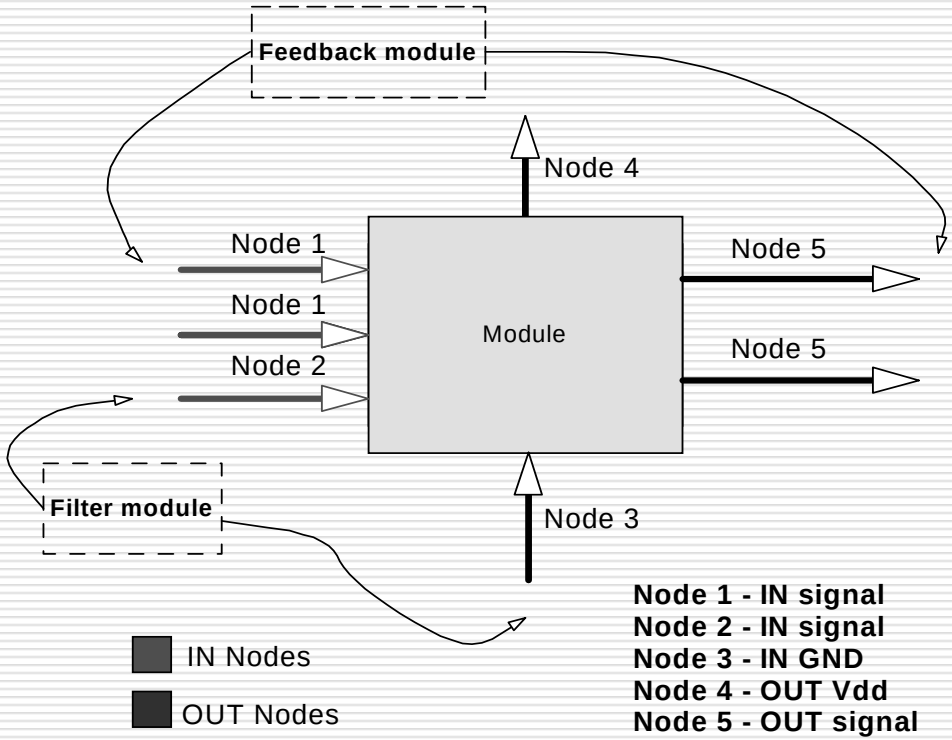


Proposal for Automated Analog design

- Step 1 – building knowledge base
  - Expressing the general logic structure of the device in terms of block structure (Ex.: Amplifier (preamp, driver, etc.), Equalizer (Low pass filter, Mid, etc), Receiver (Antenna, preamp, detector, etc)
  - Knowledge representation in terms of properties what kind or role is playing (Ex.: current amplifier, frequency rejecter)
  - Explicitly description the IN nodes and OUT nodes
  - Explicitly description additional modules for proper work. (Ex.: needs power supply, needs filter, needs feed back)
  - Explicitly description the all properties of these nodes.

Proposal for Automated Analog design

- Module with 5 nodes 2 in, 1 out and 2 power
- Distinct representation of feedback module can allow more flexibility, at the expense of more computations



Proposal for Automated Analog design

- Step 2 – formulating and partitioning the problem
  - Expressing the problem according ontology terminology (Ex.: Power supply, 10W, 5V, nonlinearity<0.001%)
  - Finding all potential general structures. (Ex.: in contrast of Audio amplifier design, instrumental amplifiers have different topology
  - Generating all possible solutions for all found structures
  - Calculating the final fitness function
  - Parametric optimization by using optimization subroutines
  - Show the result, or SPICE simulation

Proposal for Automated Analog design

- Step 3 – if solution is not obtained
  - Generate partial solution
  - For all critical modules perform evolutionary techniques to obtain better solution regarding the rules in the ontology

$$F^1 = W_1 P_1^1 + W_2 P_2^1 + ... W_n P_n^1$$

$$F_{BEST} = \min(F^i) \mid F^i < F^{REQ}$$

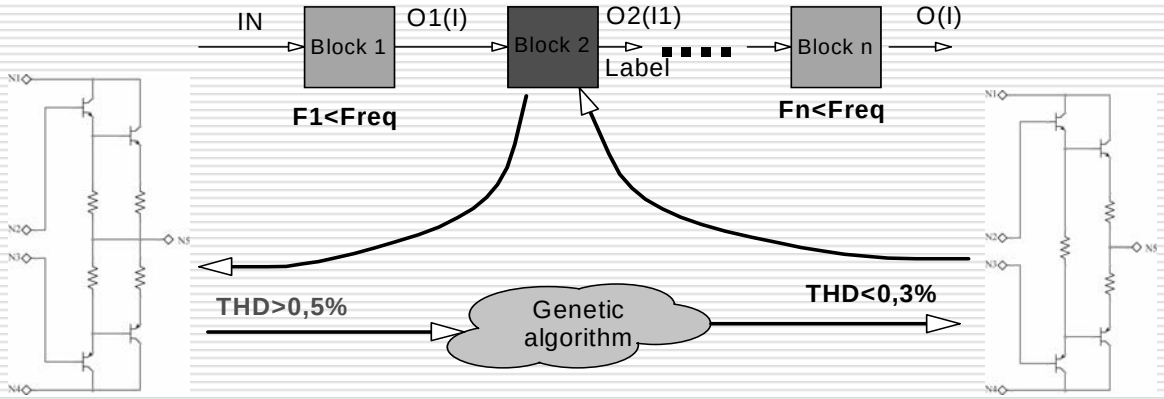
$$F^{REQ} = W_1 P_1^{REQ} + W_2 P_2^{REQ} + ... W_n P_n^{REQ}$$

$$\text{IF } F_{MIN}^I > F^{REQ} =>$$

$$(W_k P_k^i) < (W_k P_k^{REQ}) -> \text{partial solution}$$

$$(W_k P_k^i) > (W_k P_k^{REQ}) -> \text{no solution}$$

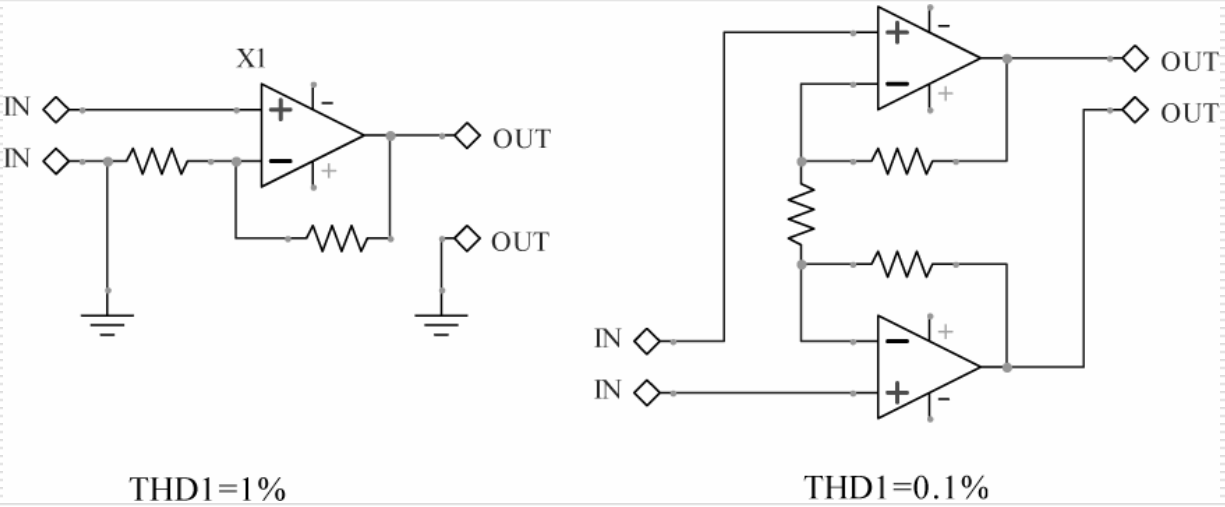
Proposal for Automated Analog design



- To avoid evolution through GA, we have to handle enormous search space
- Obtained structure by GA can be achieved by adding more attributes in the ontology representation or further module partitioning (Ex. If we can break up the above driver)

Proposal for Automated Analog design

- Step 3 – evolutionary technique in higher level
  - We apply the same variation but in module level of topology



Contributions

- Based on my present research unique features are:
  - Hierarchical knowledge representation of all assets in analog design
  - Keeping the wide circuit representation with “scalable” property – obviously the most crucial part
  - “improved” evolutionary technique by using predefined rules for cross mutation
  - Evolutionary approach in higher level of abstraction

Further work

- I have to chose domain to work out my proposal. I plane to explore Amplifier design
- I need as much as possible analytic expressions to perform element calculation
  - Big chapter of this proposal was omitted or so called "Geometric programming", due to unexpected results
  - I'm still looking for some optimization methodology (probably some nonlinear programming)
- I need to specify:
  - How would be represented the "0-level ontology"
  - How would be performed GA techniques

Thank you!

Wagner's music is better than it sounds.

Mark Twain