Almaden, May 1999 (adapted from IPMU' 92, Palma de Mallorca, July 6-10, 1992)

A global tree quality measure and its use for pruning

Louis WEHENKEL
Department of Electrical Engineering - University of Liège

Contents

Spirit of presentation

Derive quality measure from Bayesian principles

Relation with MDL principle

Relation with other stuff in DT induction

Analyse quality measure and derive pruning algorithm(s)

References on which this work is based

J. Rissanen (1978, 83), R. Sorkin (83), J. Segen (85), CART (85), P. Cheeseman (88)

References of other related work

Quinlan & Rivest (89), W. Buntine (90,92), J. Rissanen et al. (1995)

1. Problem formulation

Learning from examples

Given a learning set of objects $LS = \{o_1, \ldots, o_N\}$, where each object is described by n attribute values $a(o) = (a_1(o), \ldots, a_n(o))$, and belongs to a known class $c(o) \in \{c_1, \ldots, c_m\}$.

$\Rightarrow$ Build a model (in the form of a DT) to infer the class of an object from the knowledge of its attribute values.

Difficulty

Generalization to unseen objects required.

$\Rightarrow$ Compromise between precision (learning set fit) and reliability (unseen objects).

$\Rightarrow$ Control of the DT complexity.

$\Rightarrow$ Stop-splitting and/or pruning.

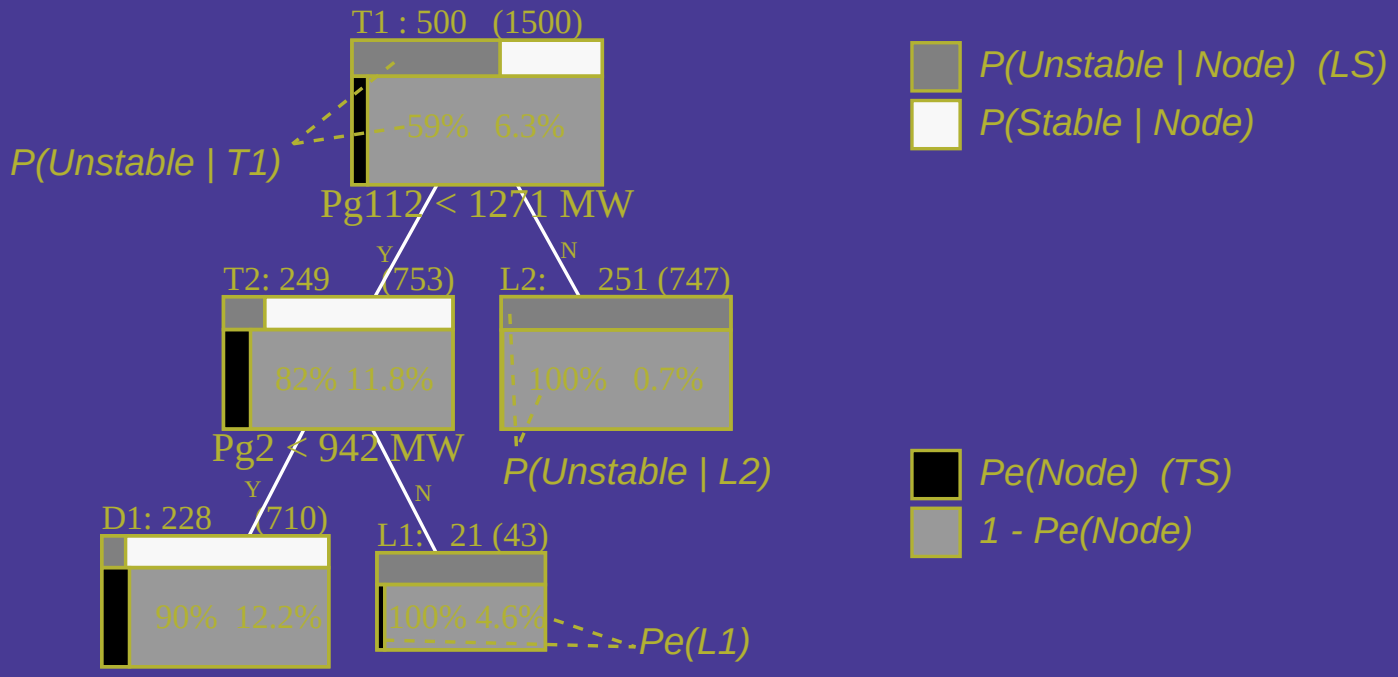
2. DT : model for conditional class-probabilities

Test nodes $\mathcal{N}_t$:

- segmentation of the attribute space $\{a(o)|\forall o\}$ into a hierarchy of subregions.

Terminal nodes $\mathcal{N}_l$ (regions) :

- conditional class-probability vectors $P(c_i|\boldsymbol{a}(o) \in \mathcal{N}_l)$.



Decision tree building (classical)

- Define a set of candidate tests (partitions) for each (type of) attribute.
- Build DT in a greedy top-down approach $\Rightarrow$ recursive partitioning
- Select tests to $\nearrow$ class purity of successors.
- Stop splitting at a (local) maximum of quality.
- If no other information is available, estimate probabilities from the LS.

NB. Everything works locally $\Rightarrow$ order of node development is irrelevant

$\Rightarrow$ most (all ?) DT induction methods use local quality measures

Classical local quality measure

Entropy in a subset (node) $\mathcal{N}$: $H_C(\mathcal{N}) = -\sum_i p(c_i|\mathcal{N}) \log_2 p(c_i|\mathcal{N})$.

$p(c_i|\mathcal{N})$: determined by frequency counts of classes of objects belonging to $\mathcal{N}$.

- The lower conditional entropy the better the tree.
- The simpler the tree the better.

$\Rightarrow$ greedy selection of tests is reasonable (ID3).

But, the approach does not account explicitly for the increase in complexity.

Not ok, if variable branching factors of candidate tests.

Unable to tell when to stop splitting.

$\Rightarrow$ Use a global quality measure explicitly taking into account complexity

3. Global quality measure for inductive inference of DTs

Account for complexity.

$\Rightarrow$Generalize to variable branching factors.

$\Rightarrow$Stop splitting criterion.

$\Rightarrow$Pruning criterion.

Bayesian formulation

Denoting by
- $c(LS)$ the observed classes in the learning set
- $P(c(LS))$ the probability of this observation as predicted by prior information
- $P(c(LS)|DT)$ the probability of this observation as predicted by the DT
- $P(DT)$ the prior probability that DT is “the good one”

We have (Bayes rule) : $P(DT|c(LS)) = \frac{P(c(LS)|DT)*P(DT)}{P(c(LS))}$

Quality measure (logs in base 2)

$$\begin{aligned} Q(DT; LS) &\doteq \log P(DT|c(LS)) \\ &= \log P(c(LS)|DT) - \log P(c(LS)) + \log P(DT) \end{aligned}$$

Discussion:

- the term $\log P(c(LS))$ does not depend on DT (see later why we keep it)
- $\log P(DT) \approx$ number of bits in optimal tree code
- $\log P(c(LS)|DT) \approx$ number of bits in optimal class code with DT
- $\log P(c(LS)) \approx$ number of bits in optimal class code without DT

Conclusion:

- $Q(DT; LS) > 0 \Leftrightarrow$ DT achieves data compression
- the larger the $Q(DT; LS)$, the better

Questions:

- How to define (compute) DT priors ?
- How to maximize quality given the priors ?

Defining priors $P(DT)$?

Possibility 1. Constant priors within a certain set of candidate DTs
($\Rightarrow$ leads to maximum-likelihood principle $\Rightarrow$ ID3)

Possibility 2. Universal priors...

$\hookrightarrow$Possibility 3. Subjective priors

Should reflect the fact that simpler trees are preferable

$\Rightarrow$ relies on the definition of complexity measure $C(DT)$

$\Rightarrow$ trees of same complexity are of same $P(DT)$

$\Rightarrow$ $P(DT) \propto \exp^{-qC(DT)}$

Because number of candidate DTs increases exponentially with complexity

Details of derivation of priors (maximum entropy principle)

Suppose that $P(DT) = f(C(DT))$.

Define :

$$P(C) \doteq \Sigma_{\{DT|C(DT)=C\}} P(DT)$$
$$N(C) \doteq |\{DT|C(DT) = C\}|$$

$$\Rightarrow P(DT) = \frac{P(C)}{N(C)}$$

For $P(C)$ different assumptions make sense (e.g.)

1. Bounded complexity : $P(C) = \frac{1}{C^{\max}}$ if $C < C^{\max}$, $P(C) = 0$ if $C \geq C^{\max}$
2. Known expected complexity : $P(C) \propto \exp^{-\beta C}$ (here $\beta \searrow$ when $E\{C\} \nearrow$).

For $N(C)$ we assume that $N(C) \propto \exp^{\gamma C}$

$\Rightarrow P(DT) \propto \exp^{-qC(DT)}$ is a reasonable choice

Defining the DT complexity measure

NB: from here on the reasoning becomes DT-specific

Complexity measure should be compatible with the recursive structure of DTs

$\Rightarrow$ Additive measure

In other words : complexity of DT = sum of complexities of subtrees

We choose :

Complexity $\doteq$ linear in number of terminal nodes

Complexity of trivial tree = 0 $\Rightarrow C(DT) =$ number of terminal nodes - 1

$\Rightarrow \log P(DT) = -qC(DT) + q_0$ (we will drop the constant q_0)

Some notation and comments before the rest

$a(o)$ denotes the attribute vector observed for object o

$\mathcal{L}(a)$ the terminal node reached by attribute vector a $(\mathcal{L}(a) \in \{\mathcal{L}_1, \ldots, \mathcal{L}_k\}, k = C(DT) + 1)$

$\mathcal{L}(o)$ stands for $\mathcal{L}(a(o))$ (depends only on $a(o)$)

$S(\mathcal{N})$ is the subset of objects in S which reach the node $\mathcal{N}$ of a DT

DT structure (topology and tests) $\equiv$ function $\mathcal{L}(\cdot) \ : a \mapsto \mathcal{L}(a) \in \{\mathcal{L}_1, \ldots, \mathcal{L}_k\}$

$c(o) \in \{c_1, \ldots, c_m\}$ denotes the class observed for object o.

$P(c(o)|DT)$ denotes the probability of $c(o)$ to happen according to the DT

DT = DT structure + class probabilities attached to each node

But, only the terminal nodes are important in terms of modeling :

$\Rightarrow P(c(o)|DT) = P(c(o)|\mathcal{L}(o))$ (probability model)

Maximization of tree quality (for a given value of q)

Choose tree structure (i.e. topology + tests) and conditional probabilities at each terminal node so as to maximize

$$Q(DT; LS) = \log P(c(LS)|DT) - \log P(c(LS)) - qC(DT)$$

Decompose the problem in two parts :

1. For a given structure determine conditional probabilities at terminal nodes
2. Determine optimal structure

Optimal conditional probabilities (easy)

- $C(DT)$ does not depend on conditional probabilities

$\Rightarrow$ for fixed structure, maximize log $P(c(LS)|DT)$ w.r.t. conditional probabilities

- Assuming $P(c(LS)|DT) = \prod_{o \in LS} P(c(o)|DT)$ (independent observations)

Gibbs $\Rightarrow$ optimal $P(c(o)|DT)$ is relative frequency of $c(o)$ in $LS(\mathcal{L}(o))$

Denoting by $LS_i = \{o \in LS | c(o) = c_i\}$ and by $N_i^j = |LS_i(\mathcal{L}_j)|$

and by $N_i = \sum_j N_i^j$, $N^j = \sum_i N_i^j$, and $N = \sum_i \sum_j N_i^j$

the optimal choice imposes $P(c_i|\mathcal{L}_j) = \frac{N_i^j}{N^j}$

$\Rightarrow$ Direct reduction of search space size

The optimal structure

- Assuming conditional probabilities are chosen optimally, we have to maximize

$$\sum_{o \in LS} \log P(c(o)|DT) - qC(DT) = \sum_i \sum_j \sum_{o \in LS_i(\mathcal{L}_j)} \log P(c(o)|DT) - qC(DT)$$

But for all $o \in LS_i(\mathcal{L}_j)$ we have $P(c(o)|DT) = P(c_i|\mathcal{L}_j)$ hence the first term is

$$\sum_i \sum_j N_i^j \log \frac{N_i^j}{N^j} = N \sum_j \frac{N^j}{N} \sum_i \frac{N_i^j}{N^j} \log \frac{N_i^j}{N^j} \doteq -N * H_{C|DT}(LS)$$

- Assuming independence and that $P(c_i) = \frac{N_i}{N}$ we have

$$-\log P(c(LS)) = -\sum_i \sum_{o \in LS_i} \log \frac{N_i}{N} = N \sum_i \frac{N_i}{N} \log \frac{N_i}{N} \doteq N * H_C(LS)$$

$\Rightarrow$ Structure optimization amounts to maximize

$$Q'(DT; LS) = N * I_C^{DT}(LS) - qC(DT)$$

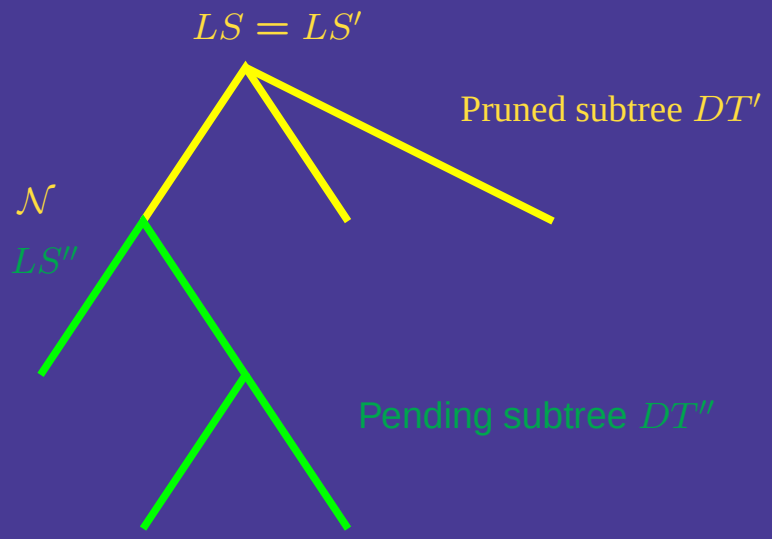
$I_C^{DT} \doteq H_C - H_{C|DT}$ is the apparent information quantity provided by DT

Main properties

Quality of trivial tree = 0. (both Info and complexity are equal to zero).

For a given DT, $[q \nearrow] \Rightarrow [Q'(DT; LS) \searrow]$

Additivity of $Q'(DT; LS)$: because $C(DT)$ and I_C^{DT} are additive



Notation :

$$DT = DT' +_{\mathcal{N}} DT''$$
$$\Rightarrow Q(DT; LS) = Q(DT', LS') + Q(DT'', LS'')$$

DT search space ($\mathcal{DT}$)

$\mathcal{DT}$ is the set of candidate DT structures considered for inductive inference.

Examples : DT growing space; DT pruning space (see later)

Assumptions about $\mathcal{DT}$ (normally hold for growing and pruning):

1. Expandability : $\forall DT \in \mathcal{DT}$, $\exists DT' \in \mathcal{DT}$ suchthat DT' can be reached by expanding a terminal node of DT (except if DT of maximal complexity)

2. Closure w.r.t. pruning : $\forall DT \in \mathcal{DT}$, all its pruned subtrees are in $\mathcal{DT}$ (in particular the trivial tree)

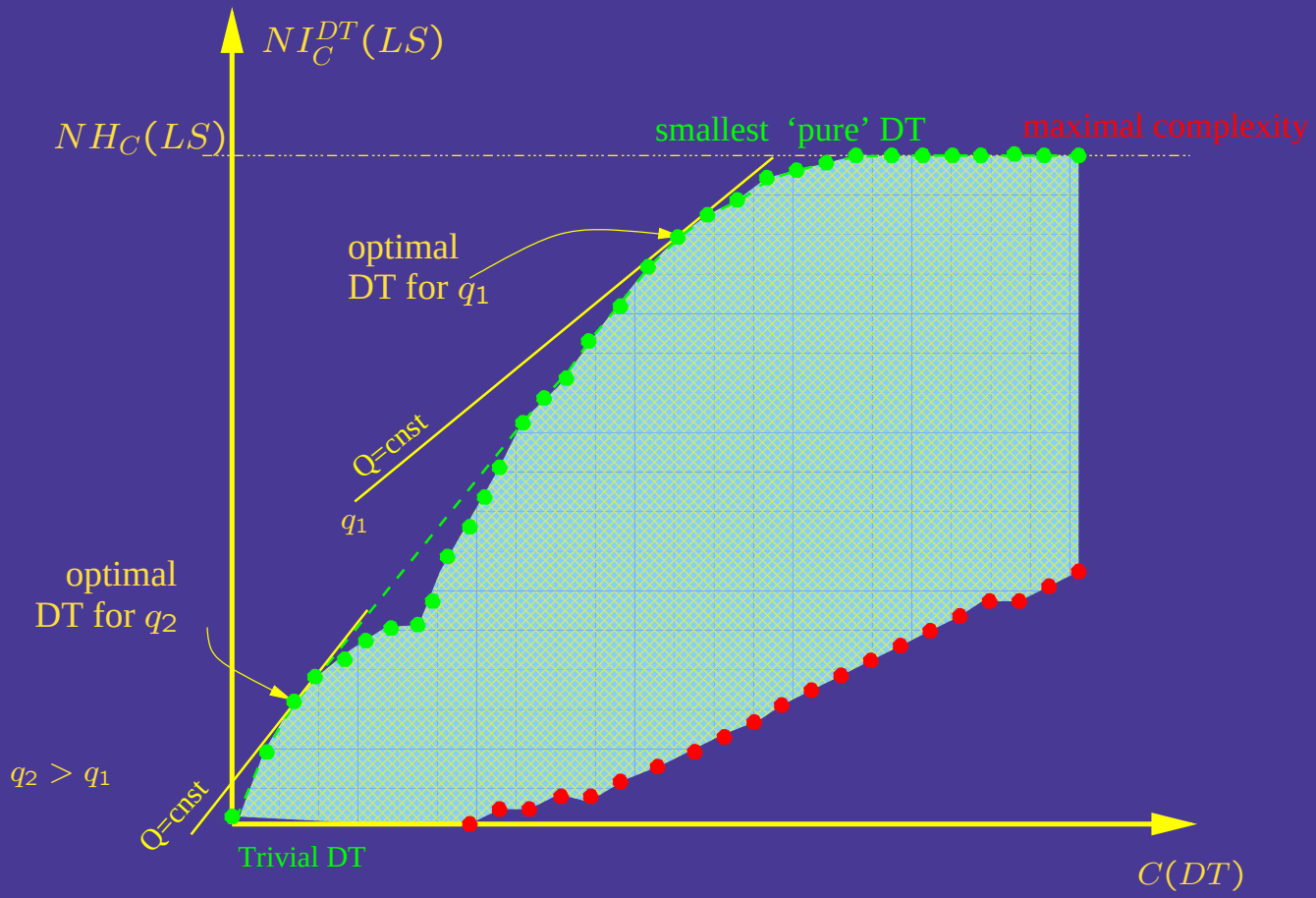
$\Rightarrow$ **Additional properties**

Decomposability : any pending subtree of an optimal tree is optimal...

Positivity : optimal tree and all its pending subtrees have positive quality

Monotonicity : if q increases $\Rightarrow$ $\left\{ \begin{array}{l} \textbf{optimal tree size decreases} \\ \textbf{optimal tree quality decreases} \end{array} \right.$

Visualization of search space $\mathcal{DT}$



Candidate DTs

- Best DT of given complexity monotone (assumption 1)
- Worst DT of given complexity monotone (assumption 2)

Maximal quality curve for different values of q (convex hull of •)

- Expansion increases Info; pruning decreases Info
- Usefull q values $\in [\underline{q}; \overline{q}]$.
- Above $\overline{q}$ optimal trees are trivial; below $\underline{q}$ optimal trees are 'pure'
- Number of critical q values upper bounded by complexity of smallest 'pure' DT.

Exploitation

1. For DT growing :

Search space is huge, and unfortunately we don’t know how to define operators to search only in the vicinity of the tradeoff curve.

But, given the good results obtained with greedy methods, there is a good chance that this type of search actually searches the neighborhood of tradeoff curve.

Stop-splitting rule : stop as soon as there is no search operator which would lead to increase in quality.

2. For DT pruning (see below)

A. if q is known : direct recursive algorithm to maximize Q.

B. if q is unknown : direct recursive algorithm to generate all DTs on tradeoff curve + cross-validation to select ‘good’ one.

4. DT pruning

Problem 1 : value of q is given

Given a DT, a LS and a value of q, extract a pruned subtree of maximal quality, by replacing the appropriate pending subtrees of the DT by terminal nodes.

Solution

Backward, bottom up recursive algorithm.

1. If trivial DT, then already “optimal”.

2. Otherwise, first prune the subtrees of its root node.

3. If resulting tree has positive quality, then it is optimal, otherwise prune it by contracting the root node.

Note : Forward pruning

Alternative algorithm.

1. Progress downwards the tree, from its root to its terminal nodes.
2. Contract a test node as soon as its own local quality is non-positive.

In the case of 2 classes and fixed branching factor, equivalent to the $\chi-$square hypothesis testing approach used for stop splitting, using the $G-$statistic.

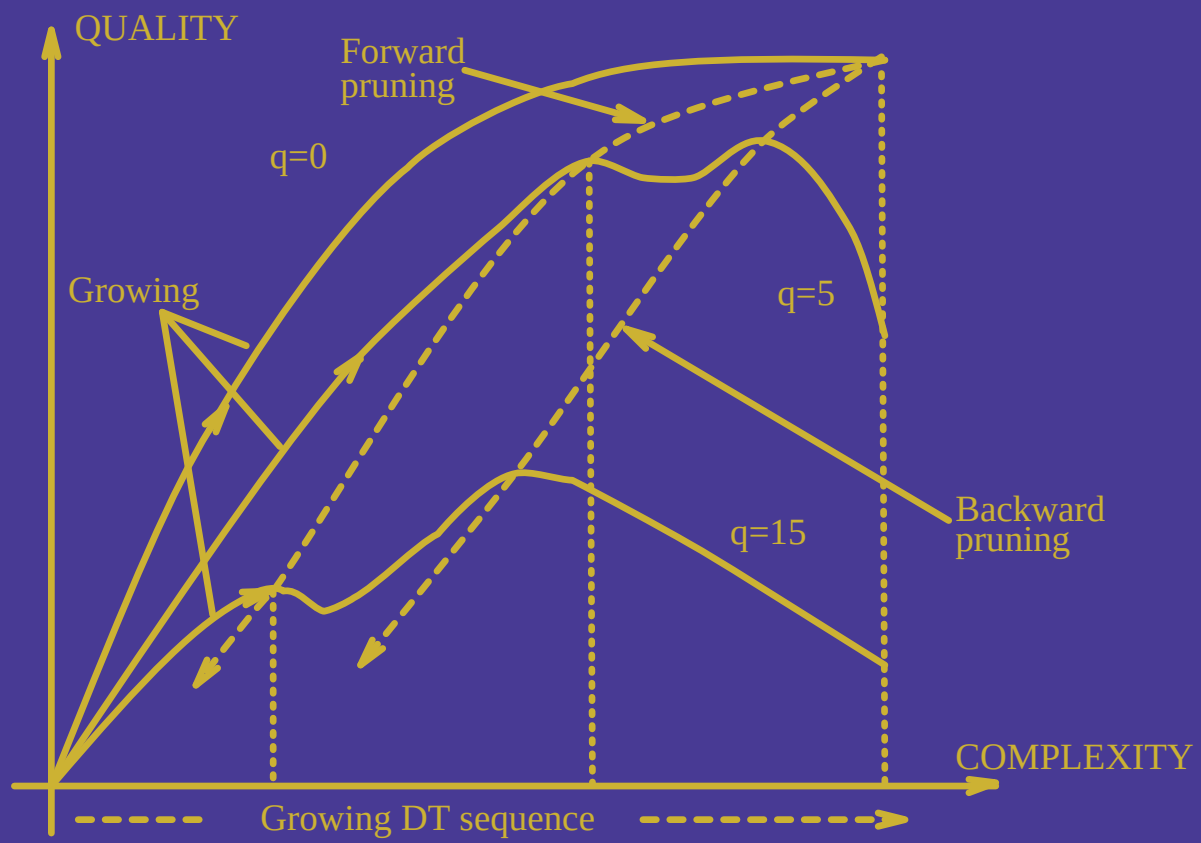
$\Rightarrow$ Provides a ground for choosing q (or α).

But suboptimal, since doesn't *look ahead*.

Often (for appropriate q values) equivalent to backward (optimal) pruning.

Effect of the value of q.

Typical quality variations during tree growing and pruning.

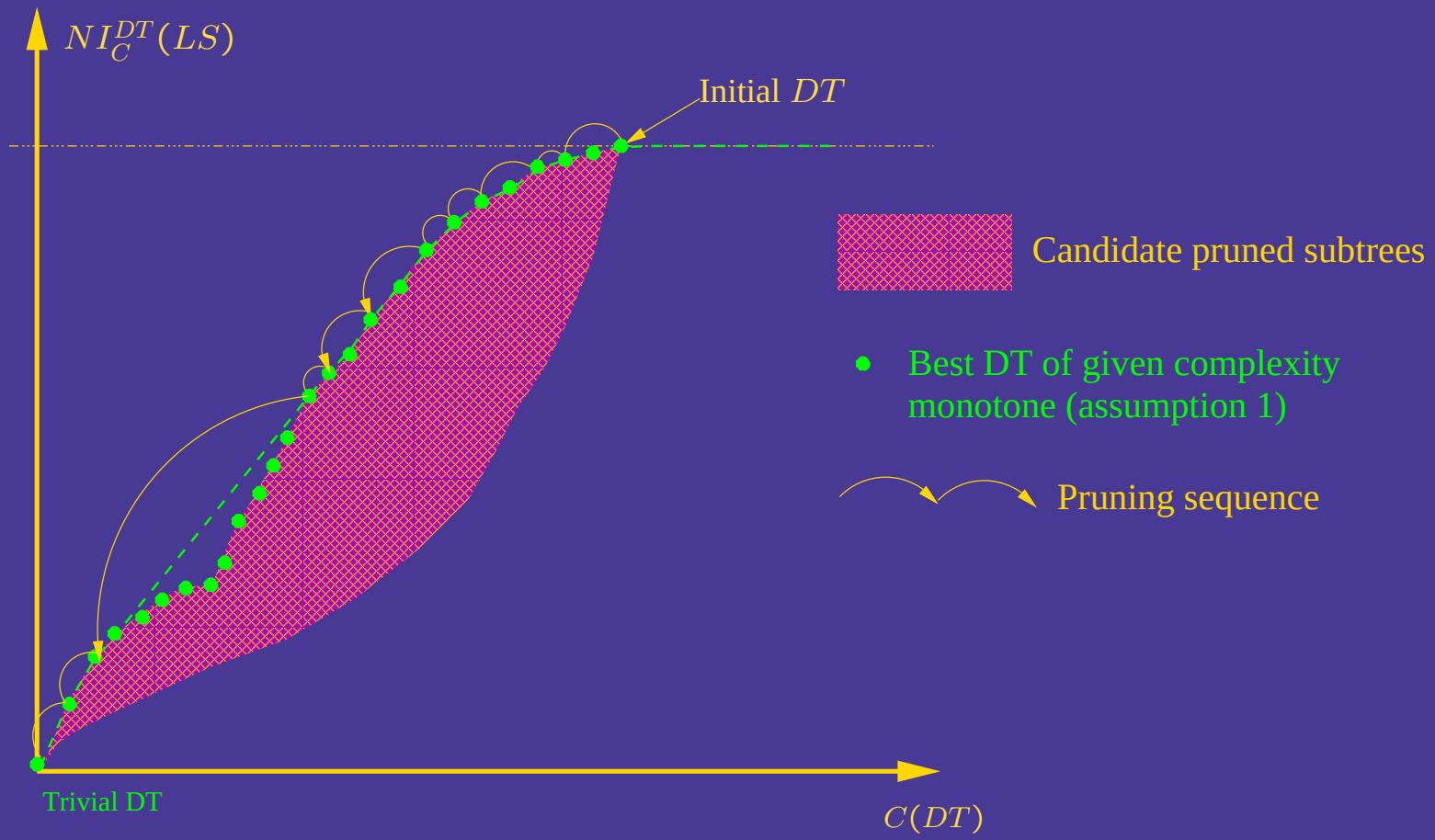


Pruning sequences

- For backward and forward pruning, and
- for every q growing from 0 to ∞,
- generate the corresponding optimally pruned DT

Problem 2 : value of q is unknown

1. Consider “special” search space : space of all pruned subtrees of DT.



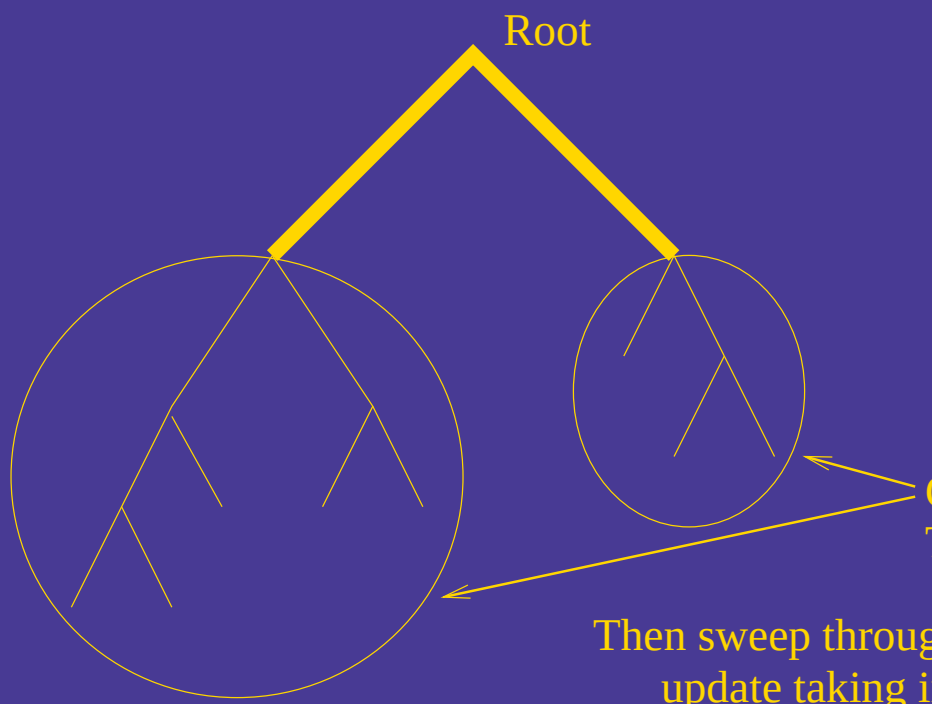
2. $\exists$ efficient recursive algorithm to generate trees on tradeoff curve
3. Test the trees on tradeoff curve using independent test set...

Properties

At most $\#\mathcal{N}_t$ different trees corresponding to sequence of critical q values.

The pruned trees form a decreasing sequence of nested trees.

Recursive generation of pruning sequence



Generate sequences recursively
Then merge (interleave critical q values)

Then sweep through sequence sorted by increasing value of q
update taking into account root, until root is contracted
replace tail of sequence by entry corresponding to trivial tree

How to choose q ?

Note. Does not depend on LS size, but is application specific.

Method 1.

Fix a priori (in the range of [10 . . . 15]). Identical to fix a non-detection risk α for hypothesis test. ($q \approx 10 \Leftrightarrow \alpha \approx 10^{-4}$)

Method 2.

On the basis of independent test set. E.g. value maximizing DT reliability w.r.t. unseen test states.

Generally method 2 is used to tune q to the application specifics. Subsequent trees are pruned using method 1, not requiring a TS.

Computational Efficiency

E.g. $<$1s on a 300MHz ULTRASPARC workstation for the generation (and testing) of the complete pruning sequence of a DT of 1000 nodes.

5. Discussion

Global quality measure :

- Close to classical entropy criteria, hypothesis tests...
- Takes into account complexity and prior information
- Allows to compare trees obtained by different methods
- Can be used for tree averaging

Additive nature :

- Nice theoretical properties
- Algorithms, analogies
- Acceptable for various types of interpretations

Generalizations :

- Other structures, other complexity measures (e.g. test-complexity...)
- Regression trees