Robust Parsing

presentation for
T-61.182 -
Robustness in Language and Speech Processing

Based on the articles
Robust Parsing And Beoynd
by Jean-Pierre Chanod
and
Robust Parsing Of Word Graphs
by Gertjan van Noord

presented by Matti Aksela

Introduction

- The need to process large quantities of natural text and speech available in electronic format boosts efforts in robust parsing

- The need for more robust methods has caused a push from traditional linguistic approaches toward statistical-based methods
  - However robustness is not about statistical vs. rule-based methods

- Robustness is about exploring all constructions humans actually produce, be they grammatical, conformant to formal models, frequent or not
  - Even rare phenomena may be common in certain domains

- Robustness is a matter of broadth and depth of linguistic analysis

Linguistic Descriptions and Robustness

- Linguistic phenomena undergo variations and distortions that are not easy to capture in core descriptions
  - linguistically sound principles do not necessarily translate into effective computational models
- Unification of lexical features presents a computationally elegant way of handling phenomena such as agreement or subcategorisation
  - however there are many situations not accounted for
- One basic requirement for a robust parser is to be able to handle both standard and non-standard phenomena

Agreement failure

- For example with standard agreement principles such as
  noun modifiers must agree in number, gender and case with the head noun they modify
  or
  the subject agrees in number with the verb,
  valid causes for agreement failure include
  - phonological constraints
  - lexical deviations
  - ellipsis, deletion
  - conflicting constraints
  - semantic constraints

- Phonological constraints
  - in French, masculine possessive adjectives may be substituted to expected feminine, if the following word begins with an vowel:
    Mon adorable chatte
    My[masc] adorable[] cat[fem]

Ungrammaticality and constraints relaxation – 1

- Traditional parsers make grammaticality judgments
  - for a robust parser attempting to determine the structure of a human-produced sentence, it is not sensible to expect either discriminating between grammatical and ungrammatical utterances
  - also humans are capable of reconstructing ungrammatical input
- Reversibility: if the same grammar is used for parsing and generating, the need for grammatical control arises.
  - for robust parsing it is largely academic
  - it can also be argued that humans can produce grammatic output while also understanding ungrammatical

Ungrammaticality and constraints relaxation – 2

- Relaxation of constraints : robust parsers accept ungrammatical data to increase coverage and hence robustness
  - the prior example could have been solved by adding the rule with vowel-following, but relaxation would also cover
    Mon chévre
    (lit.) My[masc] goat[fem] (= my goat cheese)

- Constraint relaxation may allow for more constructions to be recognized than one initially intended

- The primary role of constraints is not to prune out ungrammatical input but to eliminate spurious, unwanted analysis when a preferred analysis holds.

- Core principles should select preferred analysis whenever they are relevant, constraint relaxation should allow for a wider range of constructions when core principles do not hold

Ellipsis

- Ellipsis is one of the most systematic ways of disrupting the application of core parsing rules
  - the deletion of any word displaces the linguistic interpretation from the explicit to implicit field
    un cing tonnes
    (lit.) a[masc,sing] five-ton[fem,plur] (a five-ton truck)
    Three pints is not enough
    A good twenty minutes
  - Similar phenomena occur frequently with denomination (when the internal syntactic structure of a denominating string is superimposed by and overall different syntactic structure)
    I never saw Gone with the wind
    Fish and chips is getting expensive

Syntactic Diversity – Local Grammars

- In corpora a massive range of marginal phenomena may be observed
  - postal addresses, money, dates, citations, . . .
- Local grammars may be used to describe such phenomena
  - however the local grammar must be triggered
- May result in specific syntactic treatment that must be embedded in the overall parser
- The morpho-syntactic properties of phrases cannot be derived by mere unification of the lexical features of their subconstituents
  - In Russian numerical adjectives govern subject noun attributes, but not adjectives (which are effected my the masculinity of the noun) *dva slona*
    two elephants[sing,gen]
    dva rozovykh slona sidiat
    two pink[gen,plur] elephants[gen,sing] are seared
    de belie sobaki sidiat
    two white[nom,plur] dogs[fem,gen,sing] are seated

Semantic and Pragmatic Constraints

- Syntactic rules can be disrupted by semantic constraints
  - collective nouns representing a group (Br. Eng)
    The team are full of enthusiasm
- Syntactic information such as subcategorization gives only partial evidence of the underlying structure and cannot be reliably used to determine deep structures

Properties of Robust Parsers – 1

- The facts behind the following properties
  1. automatic description of syntactic structures is constrained by the limited background knowledge available – typically structures requiring a full semantic interpretation can be only partially analyzed
  2. the wide variety of linguistic phenomena appearing in unrestricted text must be taken into account, core principles are simply not sufficient. For this robust parsers must allow for a tractable relaxation mechanism

- The analysis incomplete and incremental
  - robust parsers do not aim at providing a full syntactic analysis in one go, but aim at resolving structures with various degrees of depth
  - the heterogeneity of the analysis depends on the nature of the syntactic constructions and the specific input to be processed

Properties of Robust Parsers – 2

- The analysis provides a partial constituent structure
  - robust parsers tend to identify parts-of-speech and a low level of syntactic category to some sequences of words
  - named chunks ($\neq$ standard phrases), may or may not represent syntactic phrases

- Construction control
  - combinatorial explosion due to describing a wide range of constructions may be a problem
  - a robust parser must also restrict the average number of parses per sentence in order to keep output usable for further processing
  - ambiguity compression through syntactic underspecification and controlled relaxation are key elements to achieve this goal
  - chances that multiple constructions occur simultaneously also grows with the size of the input

Example: PLNLP – 1

- The constructivist approach
  - based on the description of large collections of syntactic patterns
- PLNLP (Programming Language for Natural Language Processing)
  - a bottom-up parser
  - strings and substrings are represented by records (collections feature-value pairs)
  - records derived from production rules with initial records being derived from lexical entries associated with minimal tokens (terminals)
  - rules controlled by logical conditions on the value of the features

Example: PLNLP – 2

- PEG (PLNLP English Grammar)
  - broad coverage through relaxed constraints : constraints not for restricting language but for identifying the most likely interpretation
  - ambiguity reduced through collapsing multiple analysis into a single conventional compact representation
  - rules mostly binary allowing greater combination and hence improving coverage through hybrid phrases
  - conditions do not require grammaticality but disallow some constructions if a more stable one can be found
  - on parse failure, a second pass is performed with specific (tracked) rule relaxations
  - on second-pass parse failure a sequence of partial optimal analysis is produced
  - in case of multiple parses, a ranking procedure is applied (P-metric)
  - results in a syntactic sketch

Example: Constraint grammars

- The reductionist approach
  - explicitly describing all syntactic constructions nearly impossible
  - it is easier to identify conditions when a construction cannot occur
  - approach builds an initial set of all possible syntactic combinations and then prunes out ones not possible
  - constraints may be sequential elimination rules or parallel constraints
  - for example, "determiner + verb" cannot occur in English

- ENGCG (ENGlish Constraint Grammar)
  - unwanted sequences discarded leading to good speed and coverage

- French Constraint Grammar
  - results in a finite-state network with each path representing an alternative analysis for the token
  - analysis concatenated to contain all possible readings

Example: Hybrid approach

- Using constructivist or reductionist approaches depending on the syntactic phenomena to be described

- IFSP (Incremental Finite-State Parsing)
  - each input token as assigned a tag representing the part of speech and some morphological information
  - transducers compiled from regular expressions adds syntactic information
  - the syntactic information may be later deleted if some constraints so require
  - each transducer performs a specific linguistic task, for example picks up the subject
  - ordering so that most standard constructions are addressed first
  - the contextual constraints taken in consideration at any given level are incrementally relaxed

Using word graphs

- A compact representation for the sequences of words that the recognition engine hypothesizes
  - as recognition is not perfect, multiple hypothesis are produced
  - selection between hypothesis may be better done at a later stage
- Parsing word graphs is not trivial
  - if all paths were parsed . . .
  - a complete analysis should be sought, but there may be many or none especially in spoken language
- It is assumed that we have a generalized parser that can find all analysis anywhere in the input word graph, which is then annotated with the analysis. The final result is the highest-scored path through the graph.

Word Graphs

- A representation of recognition results
  - state represent points in time
  - transitions represent possibly uttered words
  - each transition has a (acoustic) score

- A word graph is a directed acyclic graph $G =< \sum, V, v_s, T, F >$ where
  - $\sum$ is a set of words (labels)
  - V is a set of states (vertices)
  - $v_s \in V$ is the start state
  - T is a set of transitions $trans(v_i, v_j, w, a)$ for a transition from v_i to v_j $(v_i, v_j \in V)$ with label $w \in \sum$ and acoustic score a
  - F is a set of finals $final(v_i, a), v_i \in V$ indicating that v_i is a final state with acoustic score a

- Recognition may result in pauses, which should be eliminated
  - grammars usually do not take into account pauses
  - reduces number of states (but increases transitions)
  - is more efficient than having pauses in lexical entries

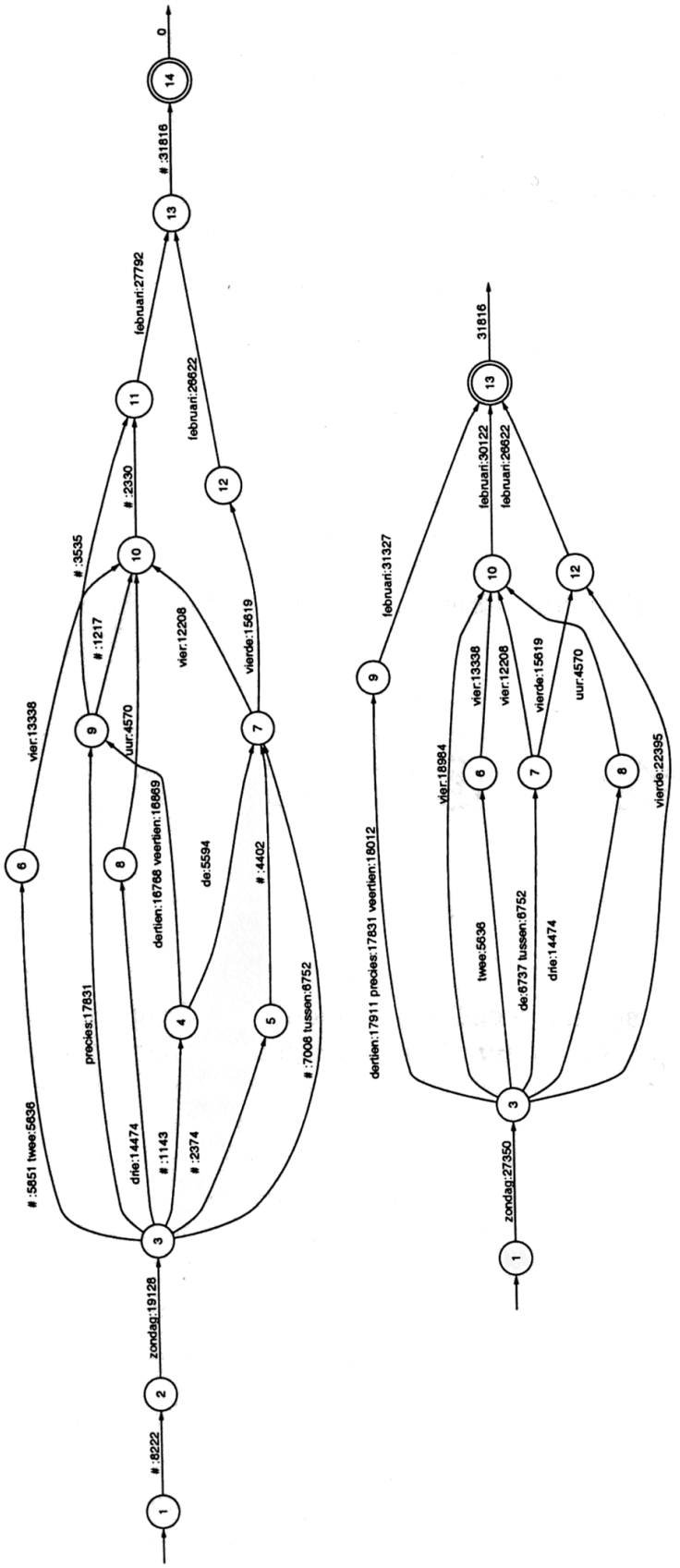
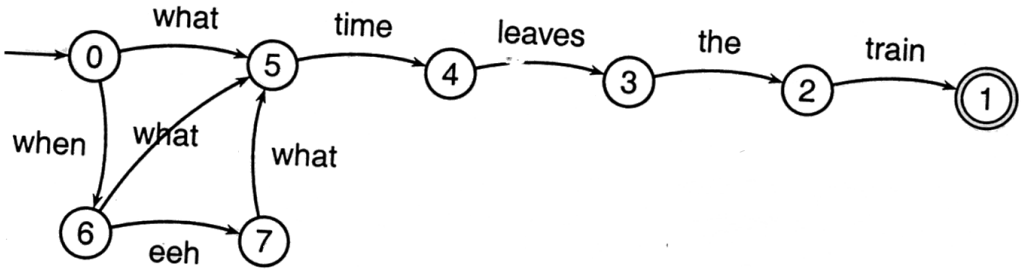


FIGURE 9.1: Word graphs for the utterance *Zondag vier februari* (Sunday Februari fourth). The special label # in the first graph indicates a pause transition. These transitions are eliminated in the second graph.

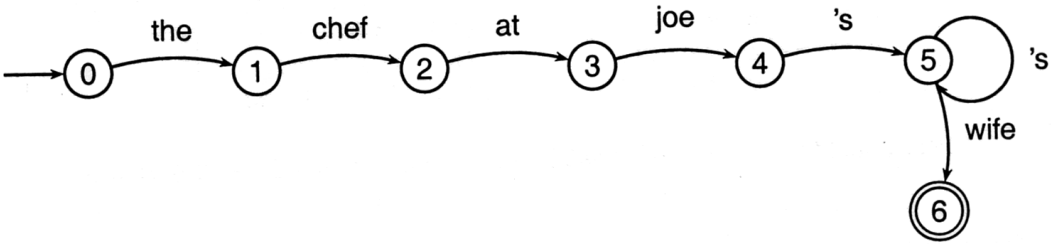
Word Graphs to FSAs

- Word graphs are Finite State Automata (FSA) with the additional constraint of no cycles occurring

- FSA is a good format for some uncertain input
  - hesitation: "When eeh what time leaves the train?"



  - haplology: "We were at a restaurant called Joe's. The food was great, but the chef at Joe's wife kept complaining about the noise we made"



  - cycles may also emerge to treat sentences with unknown parts of unknown words

Intersection of a CFG and a FSA

- Parsing can be viewed as calculating the intersection between a FSA and a CFG (Context-Free Grammar)
  - the result will be a CFG
  - the grammar generates exactly he strings in the intersection and assigns parse trees for them, resulting in a compact representation of all parse trees, a parse forest.

- Creating a FSA parser
  - replace string positions with state names
  - complexity becomes cubic in the number of states
  - can be applied to any parser

Intersection of a DCG and a FSA

- Definite Clause Grammars (DCG)
  - an example of constraint-based grammars widely used in linguistics
  - instead of atomic symbols employs first-order terms of arbitrary complexity, allowing for the representation of several linguistic phenomena as well as the construction of semantic representations
- Parsers can be converted similarly as before
- Problem: the recognition problem for DCGs is undecidable
  - this implies that it is impossible to define a parser that terminates for all inputs
  - can be circumvented by using DCGs that can be parsed, for example restricting to DCGs whose context-free skeleton doesn't contain cycles
  - most parsing algorithms terminate for each string that has a finite number of possible derivations – off-line parsability ensures this

Intersection of acyclic FSA and off-line parsable DCG

- Since word graphs are acyclic, cyclic FSA need not be considered
  - abandoning techniques presented before
- Verifying whether the intersection of an acyclic FSA and an off-line parsable DCG is empty reduces to checking whether the DCG derives any one of a finite number of strings and is hence decidable.
- Parsers can again be converted as before

Presented robust method – 1

- Ideally the parser find a path that can be assigned an analysis according to the grammar that covers the whole utterance, but this is not always possible due to
  - recognizer errors
  - linguistic constructions not covered in the grammar
  - irregularities in the utterance
- Even when no full analysis is possible, useful information can often be extracted
- With no full analysis, use something like concept spotting;
  1. grammar defined so that each maximal projection (S,NP,PP) can be analyzed as a top category
  2. the parser must find all instances of the top category anywhere in the graph
  3. an search algorithm to find a non-overlapping sequence of top categories
- The parser annotates the word graph with gained information

Presented robust method – 2

- For finding the optimal path weighting criteria must be defined:
  - Acoustic score (from speech recognizer) (a added to c_3)
  - Number of skips should be minimized to prefer maximal projections (c_1)
  - Number of maximal projections should be maximized to prefer more extended linguistic analysis (c_2)
  - $n$gram statistics (probabilities) (tri added to c_4)
  - Weights must be uniform

- For trigram weight updating:

$$uw(< c_1, c_2, c_3, c_4 >, ((v_i, w_0w_1), (v_j, y), x, a, q) =$$
$$\begin{cases} < c_1 + 1, c_2, c_3 + a, c_4 + tri(w_0w_1x) > & \text{skip edges} \\ < c_1, c_2 + 1, c_3 + a, c_4 + tri(w_0w_1x) > & \text{category edges} \\ < c_1, c_2, c_3 + a, c_4 > & \text{stop edges} \end{cases}$$

and use $total$ to define the order (k_{nlp}, k_{wq} constants):
$total(< c_1, c_2, c_3, c_4 >) = c_4 + k_{nlp} * (c_1 + c_2) + k_{wq} * c3$

Presented robust method – 3

- The robustness component can be characterized as a search in the annotated word graph
  - the goal is to find the best path from v_0 to v_{n+1}
  - a well-known graph search problem : finding the shortest path in a directed acyclic graph with uniform weights

- A search algorithm: DAG-SHORTEST-PATH
  1. sort vertices topologically
  2. in this order process an array with records for each state k the weight associated with the best known path from v_0 to v_k. Also the state history for the best known paths are stored. For each edge update the arrays if a better path to a vertex has been found.
  3. The best sequence of edges has been found, and it is trivial to extend to finding P best paths

Presented robust method – 4

- Especially using trigrams is computationally very expensive, hence filtering is needed.

- First filtering: make two passes where in the first only P best paths (judged by n-gram and acoustic scores) are kept. Second pass uses the full method.
  - B-P for bigram filtering (trigram processing still)
  - T-P for trigram filtering (trigram processing still)

- Second filtering
  - even with filtering too slow
  - define a variant of the search like beam searching
  - "beam" b defines the maximum number of paths associated with a given vertex
  - with n-gram weight no longer guaranteed to find the solution, as the weights are not uniform
  - complexity linearly dependent on b
  - M,b for method M (B-P or T-P)

Experiments

- Test performed on the OVIS2 test set
- Test set has 1000 word graphs

	Input	Pause removed
graphs	1000	1000
trans	48215	73502
states	16181	11056
words	3229	3229
t/w	14.9	22.8
max(t)	793	2943
max(s)	151	128

- Evaluation criteria:
  - Word accuracy $WA = 1 - d/n$,
    where d is the Levenshtein distance between words
  - Concept accuracy
    $CA = 1 - (SU_s + SU_i + SU_d)/SU$,
    where SU is the total number of semantic units and SU_s, SU_i, SU_d the numbers of substitutions, insertions and deletions needed to transform the analysis semantic units

Results – 1

- Speech method WA 69.8%
  - only takes into account acoustic scores, no language model
- Possible method WA 90.5%
  - the best path is always used
  - a natural upper bound
- Results for filtering methods

TABLE 9.3: Results for the subset of 822 word graphs which have at most 100 transitions.

Method	CPU mean msec	CPU max sec	WA %	CA %
B-1	16	0.1	90.9	88.2
B-2	20	0.2	91.8	90.4
B-4	26	0.3	92.8	91.2
B-8	34	0.4	93.2	91.9
B-16	45	0.6	93.2	92.0
T-1	19	0.1	92.8	91.0
T-2	24	0.2	93.0	91.9
T-4	30	0.3	92.8	91.6
T-8	40	0.5	93.2	92.0
T-16	53	0.8	93.2	92.0
nlp	76	2.7	93.2	92.0
no robust	54	1.6	57.1	64.7

TABLE 9.4: Results for full test set: filtering methods.

Method	CPU mean msec	CPU max sec	WA %	CA %
B-1	38	1.1	81.3	78.9
B-2	56	2.2	82.4	80.9
B-4	87	4.4	83.4	81.8
B-8	148	9.4	83.9	83.0
B-16	300	24.0	84.4	83.6
T-1	168	22.1	84.5	82.4
T-2	320	47.8	84.9	83.4
T-3	449	67.3	85.0	83.4
T-4	638	98.8	85.0	83.4
T-10	1801	339.6	85.0	83.5

Results – 2

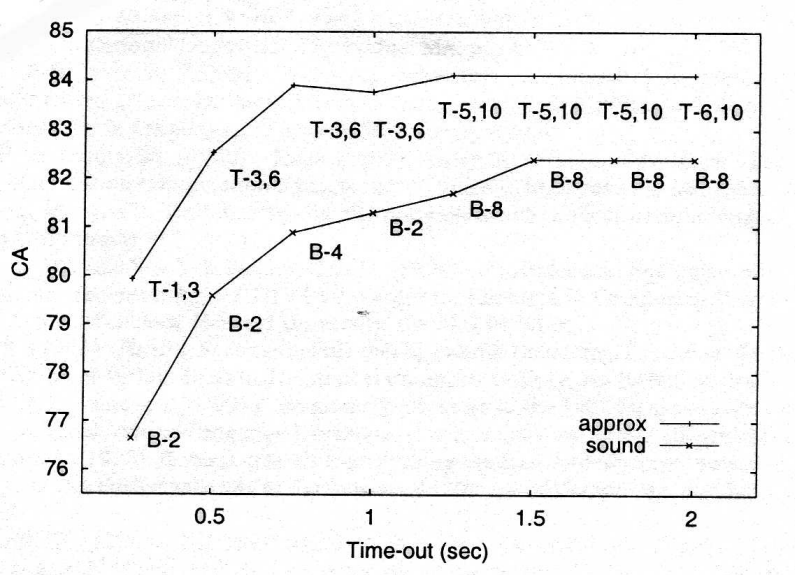
- Results for approximation methods

Method	CPU mean msec	CPU max sec	WA %	CA %
T-1,1	26	0.4	79.6	78.1
T-1,2	30	0.6	82.2	80.6
T-1,3	33	0.8	83.7	82.1
T-1,4	37	1.0	84.0	82.3
T-1,5	40	1.2	84.0	82.3
T-1,10	58	2.7	84.3	82.4
T-1,20	102	6.5	84.4	82.4
T-1	168	22.1	84.5	82.4
B-8,8	73	2.2	83.7	82.8
B-8,12	89	3.0	83.8	83.0
B-8,16	105	4.3	83.8	83.0
B-8,20	125	5.8	83.9	83.0
B-8	148	9.4	83.9	83.0

Method	CPU mean msec	CPU max sec	WA %	CA %
T-3	449	67.3	85.0	83.4
T-3,3	45	1.1	84.4	83.1
T-3,6	56	1.6	84.9	83.7
T-3,10	71	2.8	84.9	83.7
T-3,20	118	7.1	85.0	83.6
T-5,5	58	1.3	84.5	83.4
T-5,10	76	2.8	84.8	83.8
T-5,15	97	4.7	84.9	83.8
T-10,10	89	2.9	84.9	83.8
T-10,20	135	7.1	84.9	83.6
T-10,30	194	12.8	85.0	83.7
T-20,20	164	8.0	84.9	83.6
T-20,40	285	19.1	85.0	83.7
T-20,60	452	37.3	85.0	83.7

- It can be seen, that using the beam search enables the use of a higher n, which is beneficial

- Comparison of the best methods



Conclusions

- Robust parsers do not focus on formal syntax, and may in that sense rather be seen as part-of-speech disabiguators
  - part-of-speech disambiguation has been proven tractable, robust and useful from the computational viewpoint
- Also for word graph parsing robustness, in being able to find the best sequence of partial parses, is effective when a complete parse cannot be found.
- In summary, for real language robust parsing would seem like an absolute necessity.