

Extracting Noun Phrases in Subject and Object Roles for Exploring Text Semantics

Ani Thomas, M K Kowar, Sanjay Sharma
Bhilai Institute of Technology
Durg, Chhattisgarh, India

H R Sharma
Chhatrapati Shivaji Institute of Technology
Durg, Chhattisgarh, India

Abstract—In tune with the recent developments in the automatic retrieval of text semantics, this paper is an attempt to extract one of the most fundamental semantic units from natural language text. The context is intuitively extracted from typed dependency structures basically depicting dependency relations instead of Part-Of-Speech tagged representation of the text. The dependency relations imply deep, fine grained, labeled dependencies that encode long-distance relations and passive information. Apart from the typed dependencies, the present work does not take the help of Noun phrase Chunking tool or Part of speech Taggers for the compound noun phrase extraction.

Keywords-semantic analysis; parsing; dependency structures; noun phrases; subject-object nouns

I. INTRODUCTION

A document usually consists of various topics. Some topics are typically described in detail by more sentences than other topics and hence may be inferred to comprise the major content of the document. But the topics that may be briefly mentioned to supplement or support the major topics are equally important. The stages of Text Mining incorporate the most trivial step of representing sentential components meaningfully. Since four decades, text miners have explored much on the semantic representations upon texts. This corpus-based computational linguistics includes Natural Language Processing tools viz. Part-Of-Speech (POS) taggers [1], Noun-Phrase chunkers [2], Semantic Parsers, Machine Translators and Text Summarizers.

Moreover, the computational linguists at times, need the Part-Of-Speech Tagging Tools and Noun-Phrase (NP) Chunker modules to extract the typical noun-phrase fragments that pose to be a significantly thematic portion in natural language representations. Yam Chu is a Support Vector Machines based NP chunking tool and fnTBL is yet another NP chunking tool framed in C++. Gradually many Natural Language Processing (NLP) groups came in the forefront that developed NLP tools, among which OpenNLP appeared as a group of open source projects based on Maximum Entropy library called as Sharp Entropy, being a C# port of Java library [3]. These Open NLP tools supported by Java MaxEnt library, include a sentence splitter, a tokenizer, a noun-phrase chunker, a parser and a entity-name finder.

II. REPRESENTATION OF DEPENDENCY STRUCTURE

The computational linguistics communities have unanimously accepted grammatical dependencies as the most agreeable conceptual representation for any free text. Dependencies are exhibited among terms lying within or in the locality of adjacent sentences, revealing term-to-term associations for complete description of a topic being discussed in those sentences. The conceptualization of dependency parse trees was a milestone in encoding useful semantic information lying in the text. Initially the parse tree constructions needed training and revision phases, so as to identify term-dependencies in a tree and to replace incorrect dependencies with correct ones. Developments of deterministic dependency parsers revealed results with good accuracy and high performance in range of hundreds of sentences per second [4, 5].

One of the biggest breakthroughs in NLP happened in 1990s that works out with grammatical structure of sentential fragments [6]. These fragments when grouped together in multi-word phrases form the subject and object of the participating verb phrase. The Stanford group of probabilistic parsers use the knowledge gained from hand-parsed sentences in order to attempt generating the most-likely analysis of sentences in test-set. This package is a Java implementation available both in optimized PCFG and lexicalized dependency parsing versions.

The Stanford NLP Group has pioneered the concept of using typed dependencies. These are the simple descriptions of the grammatical relationships in a sentence and around its sentential neighborhood of relevant context. The salient feature that elucidates its popularity is the fact that typed dependencies are defined uniformly

every pair of words, means 1-gram phrases in text-miming terminology, rather than directly hitting phrase-structure representations, that have long dominated the minds of computational linguistic community [4]. These dependencies can be well understood without nay deep linguistic expertise for carrying out mining tasks of Ontology learning, Machine translation of text, Text Summarization, etc. however, an intermediate step between the two emerges out to be the contextual relation extraction. These semantic representations have already been studied in the form of causative relation patterns i.e. <Noun Phrase, Verb, Noun Phrase> [8]. Even RelEx Tool has been able to extract semantic relations in the format, <Subject, Predicate, Object> from the syntactically parsed representations obtained from four NL Parsers as a trial namely, Stanford, Open NLP, Link Parser and MiniPar [6, 9].

III. THE PROPOSED TECHNIQUE

The conversion strategy works on the finite set of dependencies containing fifty-five grammatical relation definitions pertaining to English Grammar, as stated by Marneffe, Manning [10]. Each grammatical relation is composed of two arguments, first the governor argument and second, the dependent argument, with parentheses pairs. The dependency definitions make use of Pen tree-bank Part-Of-Speech tags and phrasal labels. It was found upon detailed survey that only a few grammatical definitions appear in enormous amount upon parsing a sentential input, while the rest of relations pose a guest appearance, indicating the change in grammatical construction of the sentence. For instance, if the same fact is conveyed in different grammatical formats as enumerated below: the set of dependencies in Fig 1, Fig 2 and Fig 3 for the following three natural language constructions consecutively.

‘Most artificial neural system models are made of individual computational elements.’

‘The individual computational elements make up the most artificial neural system models.’

‘The individual computational elements that make up the most artificial neural system models are rarely called artificial neurons.’

Note the Stanford dependencies generated for each of the above parsed sentences carry word-position numbers along with their arguments.

```

advmod(models-5, Most-1)
amod(models-5, artificial-2)
nn(models-5, neural-3)
nn(models-5, system-4)
nsubjpass(made-7, models-5)
auxpass(made-7, are-6)
prep(made-7, of-8)
amod(elements-11, individual-9)
amod(elements-11, computational-10)
pobj(of-8, elements-11)

```

Figure 1. Typed-dependencies of sentence 1

```

det(elements-4, The-1)
amod(elements-4, individual-2)
amod(elements-4, computational-3)
nsubj(make-5, elements-4)
prt(make-5, up-6)
det(models-12, the-7)
advmod(artificial-9, most-8)
amod(models-12, artificial-9)
nn(models-12, neural-10)
nn(models-12, system-11)
dobj(make-5, models-12)

```

Figure 2. Typed-dependencies of sentence 2

```

det(elements-4, The-1)
amod(elements-4, individual-2)
amod(elements-4, computational-3)
nsubj(neural-11, elements-4)
rel(make-6, that-5)
rcomod(elements-4, make-6)
prt(make-6, up-7)
det(artificial-10, the-8)
advmod(artificial-10, most-9)
dobj(make-6, artificial-10)
nn(models-13, system-12)
nsubjpass(called-16, models-13)
auxpass(called-16, are-14)
advmod(called-16, rarely-15)
ccomp(neural-11, called-16)
amod(neurons-18, artificial-17)
dobj(called-16, neurons-18)

```

Figure 3. Typed-dependencies of sentence 3

A. Removal of non-semantic constituents

The idea is to generate the noun phrases from the typed dependencies obtained from Stanford Parser. A noun phrase (NP) may contain specifiers and qualifiers and the specifiers in turn may contain determiners. In the different classes of determiners, the articles and demonstratives are not considered as important in finding text semantics which is shown as $\text{det}(X, Y)$ in Fig. 1, 2 and 3. The authors propose the deletion of det dependencies as the first step in noun phrase extraction. But at the same time, the quantifying determiners (some, all etc) holds $\text{predet}(X, Y)$ relations and these depend on the domain for removal or consideration in the noun phrase extraction. If the domain considered for semantics is textual question answering, then predet may play a significant role. On the other hand, the quantifying determiners can be neglected in the case of text summarization. Now, the filtered dependencies can be shown in figure 4 after the removal of determiner grammatical constituents of the sentential fragments illustrated in figure 3.

```

amod(elements-4, individual-2)
amod(elements-4, computational-3)
nsubj(neural-11, elements-4)
rel(make-6, that-5)
rmod(elements-4, make-6)
prt(make-6, up-7)
advmod(artificial-10, most-9)
doj(make-6, artificial-10)
nn(models-13, system-12)
nsubjpass(called-16, models-13)
auxpass(called-16, are-14)
advmod(called-16, rarely-15)
ccomp(neural-11, called-16)
amod(neurons-18, artificial-17)
doj(called-16, neurons-18)

```

Figure 4. 'nn' relation treated typed dependencies

B. Formation of noun phrases

The typed dependencies obtained from Stanford Parser very well semantically analyses the sentences and relates the neighboring sentences. Taking these as input, the noun phrases can be extracted for processing any text document. For each noun phrase, its semantic head is marked. If the noun phrase is complex, the right most noun is identified as the head which holds for almost all noun phrases in English. Hence, the typed dependency noun compound modifier (nn) is first treated which depicts any noun that serves to modify the head noun. The typed dependency from figure 3, nn(models-14, neural-system-13) is joined to make the compound noun phrase “neural-system models”. Hence, all the existing head noun “models-14” in figure 3 will be replaced by “neural-system models -14” and the nn dependency will be removed. But there are cases when two or more dependencies occur consecutively. The picture is made clear from figure 1 where both nn dependencies: “nn(models-5, neural-3), nn(models-5, system-4)” form a noun phrase “neural system models “ by removing the two nn dependencies and replacing the compound noun in place of “models-5”.

Sentences / Contextual Role	Text (fig.1)	Text (fig.2)	Text (fig.3)
Subject Role	Individual computational elements	Individual computational elements	Individual computational elements, Artificial neurons
Object Role	Most artificial neural system models	Artificial neural system models	Most artificial, System models

The next process is to consider the amod(adjectival modifier) which specifies any adjectival phrase that serves to modify the meaning of an NP. A similar treatment as that of nn dependencies is given to a set of amod dependencies: “amod(elements-7, individual-5), amod(elements-7, computational-6)” as shown in figure 3. Since there are two consecutive amod dependencies, the noun phrase extracted will be “individual computational elements” which will further replace all “elements-7”. An advmod (adverbial modifier) is an adverbial phrase that serves to modify the meaning of the word. If an nn or amod dependency is preceded by an advmod dependency, then that advmod dependency is also treated in the same manner as *nn* and *amod* dependencies. If an *nn* or *amod* dependency contains a conjugation or cc consisting of and /or relations between two *nn* dependencies, then that has to be given the double *nn* treatment.

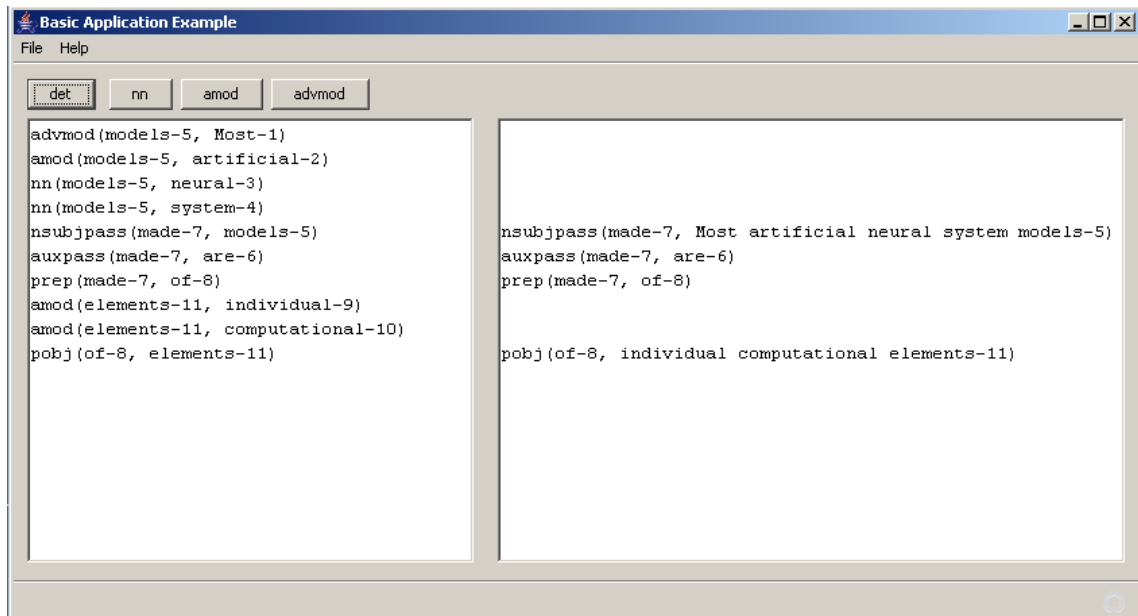
Some researchers [11] only identify the head of noun phrase but do not consider the word order for matching. This approach emphasizes upon the importance of word order as it facilitates the systems focusing on total and partial match strategy. For each combination of *nn*, *amod* or *advmod* existing adjacently, first *nn* treatment takes place followed by *amod* and then by *advmod* dependencies. Taking another set of dependencies, “advmod(models-14, most-11), amod(models-14, artificial-12), nn(models-14, neuralsystem-13)”, after treating all of them will create a noun phrase, “most artificial neural system models” and consequently replace all models in “models-14” with the above said noun phrase. The above mentioned dependency treatments have been implemented in java programming language. The noun phrase extraction snapshot of the experimented

sentences from its typed-dependencies shown in Fig 1, Fig 2 and Fig 3 is finally illustrated in Fig 5, Fig 6 and Fig 7 respectively. Further, dependent arguments for ‘*num* (number modifier)’, ‘number (element to compound number)’, ‘*neg* (negation modifier)’ relations too can be appended to grow the n-gram noun phrases in n-dimensions which are not dealt with, in the present scenario.

C. Identification of Subject and Object roles

As obviously derived from the rules of English Grammar, every sentence has one noun phrase existing in subject role and a similar phrase in object role. Although, the existence of object phrases totally depends upon the participating transitive and intransitive verbs. Stanford dependencies define the relations, namely *nsubj* (nominal subject), *nsubjpass* (passive nominal subject) and *rcmod*(relative clause modifier) to highlight the noun phrases in subject role. These appear voluntarily irrespective of any type of grammatical construction. The dependencies *dobj*(direct object) and *pobj*(object of a preposition) occurring most often depict the noun phrases in object role. The formation of the noun phrases along with the above dependencies for subject and object roles help to find semantics in natural languages to bear on real world tasks[12, 13].

TABLE I. GENERATED NOUN PHRASES IN SUBJECT AND OBJECT ROLES



Subject Role Dependencies	Object Role Dependencies
<code>advmod(models-5, Most-1)</code>	<code>nsubjpass(made-7, Most artificial neural system models-5)</code>
<code>amod(models-5, artificial-2)</code>	<code>auxpass(made-7, are-6)</code>
<code>nn(models-5, neural-3)</code>	<code>prep(made-7, of-8)</code>
<code>nn(models-5, system-4)</code>	
<code>nsubjpass(made-7, models-5)</code>	
<code>auxpass(made-7, are-6)</code>	
<code>prep(made-7, of-8)</code>	
<code>amod(elements-11, individual-9)</code>	
<code>amod(elements-11, computational-10)</code>	
<code>pobj(of-8, elements-11)</code>	

Figure 5. The noun phrases formed from typed-dependencies of sentence 1

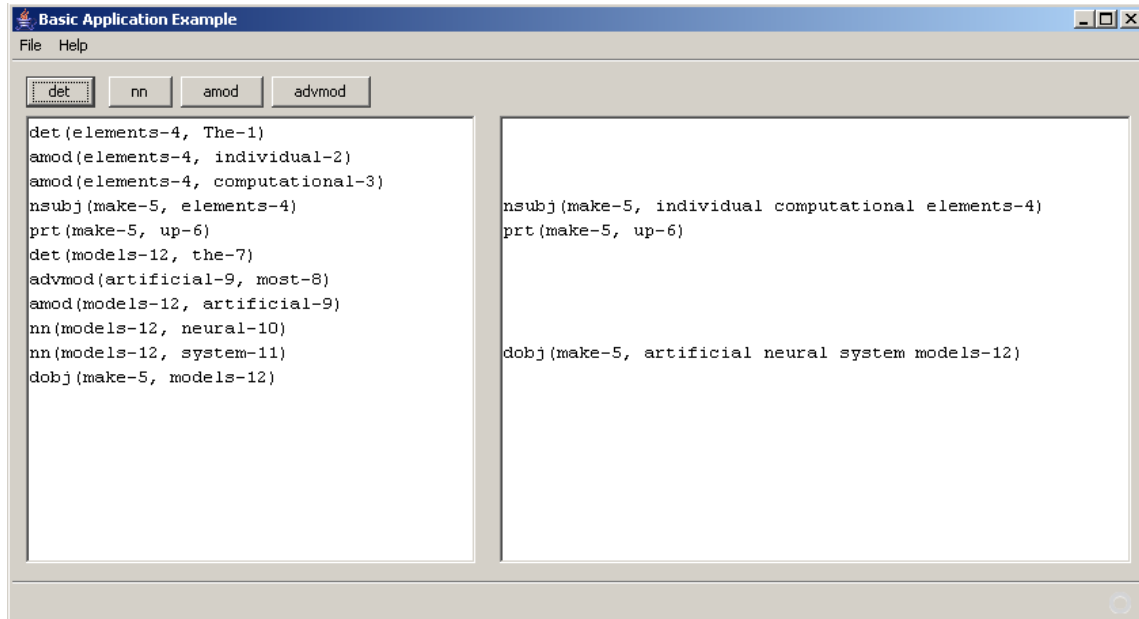


Figure 6. The noun phrases formed from typed-dependencies of sentence 2

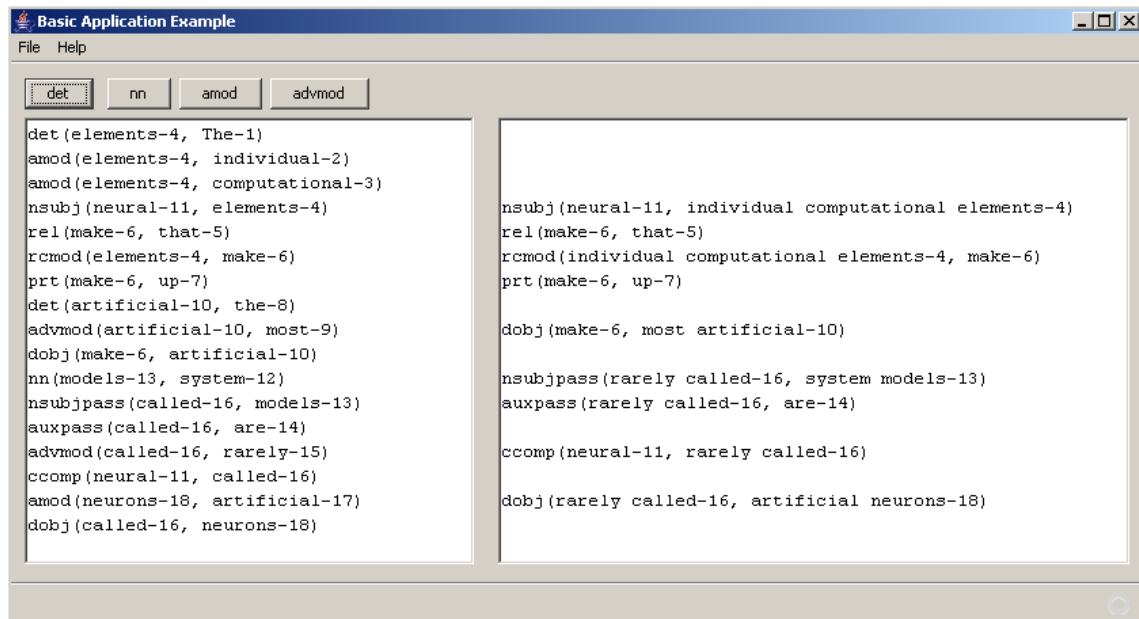


Figure 7. The noun phrases formed from typed-dependencies of sentence 3

IV. CONCLUSION

The current proposal gives a thought to the semantic analysis of any document irrespective of whether the domain is question-answering, summarization or categorization, the extraction of noun phrases plays a significant role. The plain text is inputted to the Stanford Parser which provides an accurate set of typed dependencies as output. The treatment of *det*, *nn*, *amod*, and *advmod* dependencies helps in the retrieval of compound nouns or noun phrases. Another set of dependencies like *nsubj*, *dobj* etc. helps in finding whether the noun phrases exist in subject or object roles. For the complete range of noun phrases, the replacement of accurate noun phrases in place of pronouns and also dealing of noun to noun relations existing in other dependencies are also being explored as a part of the pursuing research.

ACKNOWLEDGMENT

The authors wish to thank the management of Bhilai Institute of Technology, Durg, India for their required encouragement and support towards the completion of the work and also Sujata Kataria for providing the fully downloaded version of Stanford parser and tagger in time for the necessary implementation.

REFERENCES

- [1] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. In *Proceedings of HLT-NAACL 2003* pages 252-259.
- [2] Philip Brooks, "SCP: Simple Chunk Parser", Artificial Intelligence Center, University of Georgia, Athens, Georgia, USA, May 2003, www.ai.uga.edu/mc/ProNTo/Brooks.pdf.
- [3] Kristina Toutanova and Christopher D. Manning. Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000), Hong Kong*.
- [4] Giuseppe Attardi, Massimiliano Ciaramita: Tree Revision Learning for Dependency Parsing. *Proceedings of HLT-NAACL 2007, Rochester, 2007*. pp 388-395.
- [5] Aoife Cahill, Michael Burke, Ruth O'Donovan, Stefan Riezler, Josef van Genabith and Andy Way (2008) *Wide-Coverage Deep Statistical Parsing using Automatic Dependency Structure Annotation*, Computational Linguistics, Vol. 34, No. 1, pages 81-124.
- [6] Katrin Fundel, Robert Küffner, Ralf Zimmer. RelEx - Relation extraction using dependency parse trees. *Bioinformatics*, vol 23, no. 3, pp. 365-371, 2007.
- [7] Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. *5th International Conference on Language Resources and Evaluation (LREC 2006)*, pp. 449-454.
- [8] Roxana Girju. 2003. Automatic Detection of Causal Relations for Question Answering. In the proceedings of the *41st Annual Meeting of the Association for Computational Linguistics (ACL 2003), Workshop on "Multilingual Summarization and Question Answering - Machine Learning and Beyond"*.
- [9] D. Rusu, L. Dali, B. Fortuna M. Grobelnik, D. Mladenic. Triplet Extraction from Sentences. In Proceedings of the 10th International Multiconference "Information Society - IS 2007" (Ljubljana, Slovenia, October 8--12, 2007), pp-218--222.
- [10] Marie-Catherine de Marneffe and Christopher D. Manning, "Stanford typed dependencies manual", September 2008.
- [11] C. Monz and M. de Rijke. Tequesta: The University of Amsterdam's Textual Question-Answering System. In Notebook papers TREC-10, 2001.
- [12] Ani Thomas, M.K. Kowar, S.Sharma: Automatic Subject Indexing: A Bootstrapping Approach through Text Mining operations, *International Journal of Engineering Research and Industrial Applications*, Vol 1, No.5, pp.163-174 (2008).
- [13] Ani Thomas, M.K. Kowar, S.Sharma: Intelligent Fuzzy Decision Making for Subjective Answer Evaluation, [online], IEEE Xplore release 2.5, DOI: 10.1109/ICETET.2008.216 (2008).

AUTHORS PROFILE



Mrs. Ani Thomas is Associate Professor in Department of Computer Applications of Bhilai Institute of Technology, Durg (C.G.) , India. She obtained her Masters degree in Computer Applications from Government Engineering College, Raipur. Mrs. Thomas is an amateur Research fellow focusing her research in exploring text mining techniques for generating automatic subjective question answering system. She has eight research papers published at various peer reviewed journals to her credit. She has also got some of her technical papers published in eight National levels and seven International level Conferences.



Prof. H. R. Sharma is a Professor in Department of Computer Science & Engineering and Principal at Chhatrapati Shivaji Institute of Technology, Durg, Chhattisgarh, India. He is a postgraduate in Applied Mathematics from Jabalpur University in 1966, completed his another post graduation course in Computer Science from Delhi University in 1991. Further, his research proceeded in the field of Mathematics in the area of Numerical Analysis & high Speed Computation" that was awarded to him from IIT Delhi in the year 1970. Having a total teaching experience for more than 38 years, he has supervised 03 Ph.D. scholars at Maharishi Dayanand University, Rohtak, Haryana , and 02 Ph.D. scholars at Pt. Ravishankar Shukla University, Raipur, Chhattisgarh in the discipline of Engineering and Technology (Computer Sciences). Professor H R Sharma is an Editor to the reputed International Journals of Computer Science and Information Technology, Electronics and Electrical Engineering and also Emerging Journal of Engineering Science and Technology.



Prof. Manoj Kumar Kowar is presently the Principal of Bhilai Institute of Technology at Durg (C.G.) India. He has total teaching experience of 23 years. His primary research interests include Modelling & Simulation of Fabrication Processes, Bio-Medical instrumentation, Fuzzy logic in Medical Expert systems and Secure Communications. He has a total of 129 Research papers published in refereed International Journal and Conference Proceedings. He is member of various prestigious professional organisations and reviewer of International Journals of repute.



Dr. Sanjay Sharma is Professor in the Department of Mathematics in Bhilai Institute of Technology at Durg (C.G.) , India. He received his Ph.D. degree from Pt. Ravi Shankar Shukla University Raipur, Chhattisgarh. Dr. Sharma's primary research interests focus on analyzing and studying the effects of Solar Radiation Pressure and other nature's forces on Interconnected Satellite system orbits. During the nineteen Years of Teaching Experience, he has published many technical papers in reputed Journals like NASA, Bulletin of the Astronomical Society of India, Indian Journal of Pure and Applied Mathematics, Bulletin of the Calcutta Mathematical Society , The Mathematics Education and so on.