Conjunctive Query Answering for Description Logics with Transitive Roles

Birte Glimm* Ian Horrocks Ulrike Sattler
University of Manchester, UK
[glimm,horrocks,sattler]@cs.man.ac.uk

1 Introduction

Existing Description Logic (DL) reasoners[1] provide automated reasoning support for checking concepts for satisfiability and subsumption, and also for answering queries that retrieve instances of concepts and roles. The development of a decision procedure for conjunctive query answering in expressive DLs is, however, still an open question. *Grounded* conjunctive queries for $\mathcal{SHIQ}$ are supported by KAON2, Pellet, and Racer's query language nRQL. However, the semantics of grounded queries is different from the usually assumed open-world semantics in DLs, since existentially quantified variables are always replaced with individual names.

None of the existing conjunctive query answering techniques [11, 9, 3, 8] is able to handle transitive roles or nominals in the query body.[2] In this paper, we present an extension of $\mathcal{SHQ}$ with a restricted form of the binder operator ($\downarrow$) and state variables known from Hybrid Logics [2], which allows to extend the rolling-up technique [11] to transitive roles. Further on, we adapt the $\mathcal{SHQ}$ tableaux algorithm [5] in order to decide conjunctive query entailment with this extended logic. We also highlight why the extension with either nominals or inverse roles makes the design of such a decision procedure much harder. Query answering for $\mathcal{SHOQ}$, i.e., $\mathcal{SHQ}$ plus nominals, can, however, be realised by a suitable guessing technique that we introduce in Section 4.

*This work was supported by an EPSRC studentship.

[1]For example, FaCT++ `http://owl.man.ac.uk/factplusplus/`, KAON2 `http://kaon2.semanticweb.org/`, Pellet `http://www.mindswap.org/2003/pellet/`, or Racer Pro `http://www.racer-systems.com/`

[2]Although the algorithm presented by Calvanese et al. [3] allows the use of regular expressions, in particular the transitive reflexive closure, in the query it has been shown that the algorithm is incomplete [6, 4].

2 Preliminaries

Let $\mathcal{L}$ be a Description Logic, C an $\mathcal{L}$-concept, and N_V a finite set of variable names with $y \in N_V$. With $\mathcal{L}_\downarrow$ we denote the language obtained by allowing, additionally, y and $\downarrow y.C$ as $\mathcal{L}$-concepts.

For an interpretation $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, an element $d \in \Delta^\mathcal{I}$, and a variable $y \in N_V$, we denote with $\mathcal{I}_{[y/d]}$ the interpretation that extends $\mathcal{I}$ such that $y^\mathcal{I} = \{d\}$. The $\mathcal{L}_\downarrow$-concept $\downarrow y.C$ is then interpreted as $(\downarrow y.C)^\mathcal{I} = \{d \in \Delta^\mathcal{I} \mid d \in C^{\mathcal{I}_{[y/d]}}\}$.

Let $\vec{y}$ be a vector of *non-distinguished variables* and $\vec{c}$ a vector of individual names. A *Boolean conjunctive query* q has the form $\langle\rangle \leftarrow conj_1(\vec{y}; \vec{c}) \wedge \ldots \wedge conj_n(\vec{y}; \vec{c})$. We call $\mathbf{T}(q) = \vec{y} \cup \vec{c}$ the set of *terms* in q,[3] and we call each $conj_i(\vec{y}; \vec{c})$ for $1 \leq i \leq n$ an atom. Atoms are either concept or role atoms: a concept atom has the form $t_1 : C$, and a role atom the form $\langle t_1, t_2\rangle : r$, for $\{t_1, t_2\} \subseteq \mathbf{T}(q)$, C an $\mathcal{L}$-concept, and r an $\mathcal{L}$-role.

Let $\mathcal{K}$ be an $\mathcal{L}$ KB, $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$ a model for $\mathcal{K}$, q a Boolean conjunctive query for $\mathcal{K}$, and $\cdot^A : \mathbf{T}(q) \to \Delta^\mathcal{I}$ an assignment in $\mathcal{I}$. We say that q *is true in* $\mathcal{I}$ and write $\mathcal{I} \models q$ if there exists an assignment $\cdot^A$ in $\mathcal{I}$ s.t. $t^A \in t^\mathcal{I}$ for every individual $t \in \vec{c}$, $t^A \in C^\mathcal{I}$ for every concept atom $t : C$ in q, and $\langle t_1^A, t_2^A\rangle \in r^\mathcal{I}$ for every role atom $\langle t_1, t_2\rangle : r$ in q. If $\mathcal{I} \models \mathcal{K}$ implies $\mathcal{I} \models q$ for all models $\mathcal{I}$ of $\mathcal{K}$, then we say that q *is true in* $\mathcal{K}$, and write $\mathcal{K} \models q$; otherwise we say that q *is false in* $\mathcal{K}$, and write $\mathcal{K} \not\models q$.

Since answering non-Boolean conjunctive queries can be reduced to answering (possibly several) Boolean queries, we consider only Boolean queries here.

3 The Rolling-Up Technique with Binders ($\downarrow$)

The rolling-up technique is used to reduce conjunctive query answering to KB satisfiability. This is achieved by reducing a query q into a concept expression C_q and by testing if adding $\top \sqsubseteq \neg C_q$ makes the KB unsatisfiable. This works well for tree-like queries. Here we show how cyclic queries for an $\mathcal{L}$ KB can be rolled-up into $\mathcal{L}_\downarrow$-concepts.

A Boolean conjunctive query q can be represented as a directed, labelled graph where the nodes correspond to the terms in the query. The nodes are labelled with the concepts that occur in the corresponding concept atoms. The edges correspond to the role atoms in q and are labelled accordingly. For example, let q_1 be the query $\langle\rangle \leftarrow x : C \wedge \langle x, y\rangle : s \wedge y : D$ and let q_2 be the query $\langle\rangle \leftarrow x : C \wedge \langle x, y\rangle : s \wedge \langle y, x\rangle : r \wedge y : D$. The query graph for q_1 is depicted on the left hand side and the one for q_2 on the right hand side of Fig. 1.

[3]For readability, we sometimes abuse our notation and refer to $\vec{y}$ as a set. When referring to a vector $\vec{y}$ as a set, we mean the set $\{y_i \mid y_i \text{ occurs in } \vec{y}\}$.

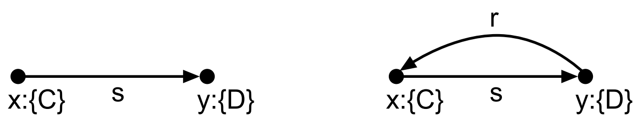


Figure 1: The query graphs for q_1 and for q_2.

Since q_1 is acyclic, the standard rolling-up technique can be used to build a concept that represents q_1: We remove y and its incoming edge and conjoin $\exists s.D$ to the label C of x resulting in $C \sqcap \exists s.D$ for C_{q_1}. A given KB $\mathcal{K}$ entails q_1 iff $\mathcal{K} \cup \{\top \sqsubseteq \neg C_{q_1}\}$ is unsatisfiable.

This reduction is not directly extendable to cyclic queries since, due to the tree-model property of most DLs, a concept cannot capture cyclic relationships. The binder ($\downarrow$), however, allows to label elements in a model with a variable, and this variable can be used to enforce a co-reference. The query concept C_{q_2} for q_2 can then be expressed as $\downarrow x.(C \sqcap \exists s.(D \sqcap \exists r.x))$. Now we have again that $\mathcal{K} \models q_2$ iff $\mathcal{K} \cup \{\top \sqsubseteq \neg C_{q_2}\}$ is unsatisfiable.

Even for cyclic queries the rolling-up for $\mathcal{SHQ}$ is now straightforward, provided that each connected component in the query graph is also strongly connected. In the presence of weakly connected components, we would need inverse roles in order to roll-up a query. Tessaris [11] shows how arbitrary conjunctive queries can be rolled-up even without inverse roles and we expect that we could combine his approach with binders and variables as well.

Unfortunately, there are no known decision procedures for expressive DLs with the binder operator. It is even known that $\mathcal{ALC}$ extended with binders and state variables is undecidable [1]. However, we observe some interesting properties for our query concepts. (1) After rolling-up, all variables occur only positively. (2) Only existential restrictions are introduced in the rolling-up. Hence, after negating the query concept and transforming it into negation normal form (giving $\downarrow x.(\neg C \sqcup \forall s.(\neg D \sqcup \forall r.\neg x))$), we have that all variables occur only negated and are only under the scope of universal quantifiers. For $\mathcal{ALC}$, these restrictions are enough to regain decidability [10].

3.1 A Tableaux Algorithm for $\mathcal{SHQ}$ Query Concepts

The tableaux algorithms for $\mathcal{SHIQ}$ [7] or $\mathcal{SHOQ}$ [5] are both capable of deciding KB satisfiability for $\mathcal{SHQ}$. For handling query concepts that may contain binders and state variables, some adaptations are necessary. For storing the bindings of variables, we modify the labels to contain tuples of the form $\langle C, B\rangle$, where C is a concept and B is a (possibly empty) set of bindings for the free variables in C. The next obvious addition is a rule for handling concepts of the form $\downarrow y.C$. Therefore, if $\langle \downarrow y.C, B\rangle$ is in the label of a node v, we add $\langle C, \{y/v\} \cup B\rangle$ and $\langle y, \{y/v\}\rangle$ to the label of v. This states that y is bound to v at v and that the free variable y in C is bound to v as well. All other existing rules have to propagate the bindings as well, e.g., the $\forall$-rule applied to $\langle \forall r.C, B\rangle$ in the label of a node v adds $\langle C, B\rangle$ to the labels of v's r-successors. The set B contains all and only the bindings for the free variables in C. Another obvious consequence is the addition of a new clash condition: If both $\langle y, \{y/v\}\rangle$ and $\langle \neg y, \{y/v\}\rangle$ are in the label of the node v, then this is a clash.

A more challenging task is the extension of the blocking condition. For $\mathcal{SHQ}$, however, we argue that we can simply ignore the bindings, i.e., if $\langle C, B\rangle$ is in the label, we consider only C in the blocking condition. This clearly ensures termination. But why does this guarantee that we can unravel a complete and clash-free completion forest into a tableau? Obviously, in $\mathcal{SHQ}$, there is no way for a node to propagate information back to its ancestors, and clashes according to the new clash condition can only occur through a cyclic structure. This is because a node v is only labelled with $\langle y, \{y/v\}\rangle$ by an application of the new $\downarrow$-rule to some concept $\langle \downarrow y.C, B\rangle$ in the label of v. Furthermore, the only way $\neg y$ can occur with v as a binding is when $\langle C, \{y/v\} \cup B\rangle$ is expanded to $\langle \neg y, \{y/v\}\rangle$ via a cyclic path back to v. This is obviously only possible among individual nodes in $\mathcal{SHQ}$ and, therefore, no clash in the tableau can by caused by unravelling. Hence, transitive roles alone are not causing major problems.

An interesting consequence is, however, that we loose the finite model property. For example, let $\mathcal{K}$ contain the axioms $\top \sqsubseteq \exists r.\top$ and $\top \sqsubseteq \neg C_q$ with r a transitive role and $\neg C_q \equiv \downarrow x.(\forall r.\neg x)$. The first axiom enforces an infinite r-chain for every individual. Normally, a finite model could contain an r-cycle instead of an infinite chain, but this would clearly violate $\neg C_q$. Hence, every model of $\mathcal{K}$ must be acyclic and therefore contain an infinite r-chain.

4 A Rolling-up Technique for $\mathcal{SHOQ}$

For the DL $\mathcal{SHOQ}$, i.e., $\mathcal{SHQ}$ extended with nominals, we propose to extend the guessing technique by Calvanese et al. [3]. In the setting considered in [3], cycles must involve individuals explicitly named in the ABox. Hence, by non-deterministically replacing variables in a cycle with individual names, the query can be rolled-up. In the presence of nominals this variable replacement is not sufficient. For example, Fig. 2 represents a model for the KB containing the axioms $\{a\} \sqsubseteq \neg C \sqcap \neg D \sqcap \exists s.(C \sqcap \exists r.(D \sqcap \exists s.\{a\}))$ and $\mathsf{trans}(s)$.

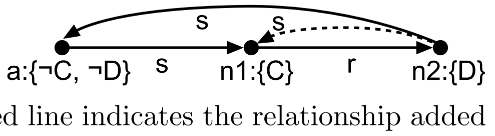


Figure 2: The dashed line indicates the relationship added due to s being transitive. Therefore, there is a cycle not directly containing the nominal a.

The query $\langle\rangle \leftarrow x : C \wedge y : D \wedge \langle x, y\rangle : r \wedge \langle y, x\rangle : s$ (see Fig. 3) would clearly be true, although a cannot be bound to either x or y. For $\mathcal{SHOQ}$, however, a cycle among new nodes can only occur due to a transitive role that provides a shortcut for "skipping" the nominal. Hence, a nominal is always at least indirectly involved in a cycle. In this case, we have only one nominal a, and we may guess that it is either in the position of x, in the position of y, or it is "splitting" the role s. In each case, we can roll-up into the nominal, obtaining three query concepts. The query is true just in case one of these is entailed by the KB. Clearly, in our example, only the concept corresponding to the third "new" guess is entailed. If we had n nominals, then we would need to try $3 * n$ guesses plus the guesses obtained by equating variables as suggested in [6].

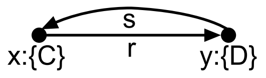


Figure 3: The original query graph.

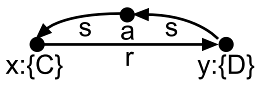


Figure 4: An alternative query graph in which the nominal a is assumed to be involved in the cycle.

Since it was, to the best of our knowledge, not even known if conjunctive query answering for logics with nominals and transitive roles is decidable, this technique is clearly valuable, although it is, due to its highly non-deterministic nature, not very practical.

5 The Challenges of Inverses and Nominals

The arguments used for the extension of $\mathcal{SHQ}$ with binders (i.e., blocking can ignore different bindings) and the extended guessing strategy for $\mathcal{SHOQ}$ cannot be used with inverse roles. Fig. 5 shows a representation of a model for the concept $\{a\} \sqcap \exists r.(\exists s.\top)$ for s a transitive and symmetric role. The query $\langle\rangle \leftarrow \langle x, x\rangle : s$ is obviously true in this model. The nominal a is, however, not even indirectly involved in the cycle.

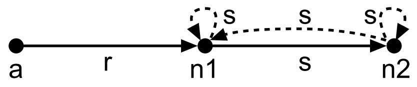


Figure 5: The dashed lines represent the additional relationships added for s being transitive and symmetric.

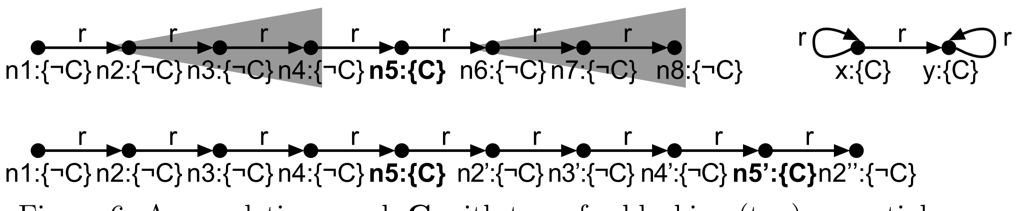
Since completion graphs are finite representations of models, the question is, how far do we have to expand a completion graph before blocking is "safe", i.e., unravelling into a tableau does not lead to a clash.

To see the same problem from another perspective, we briefly sketch another possible query answering algorithm in the style of CARIN [9]. CARIN provides a conjunctive query answering algorithm for $\mathcal{ALCNR}$ and is based on the idea that, if the query is true in each model $\mathcal{I}$ of $\mathcal{K}$, then there is a mapping $\sigma_\mathcal{I}$ from terms in the query to the individuals in $\mathcal{I}$ s.t., if $x : C$ is a concept atom ($\langle x, y\rangle : r$ is a role atom) in q, then $\sigma_\mathcal{I}(x) \in C^\mathcal{I}$ ($\langle \sigma_\mathcal{I}(x), \sigma_\mathcal{I}(y)\rangle \in r^\mathcal{I}$). We call such a mapping a q-mapping. A q-mapping ϕ for completion graphs is defined s.t., if $x : C$ is a concept atom in q, then C is in the label of $\phi(x)$ and, if $\langle x, y\rangle : r$ is a role atom in q, then $\phi(y)$ is an r-descendant of $\phi(x)$, where r-descendant implicitly closes the transitive roles. Since ϕ is purely syntactic, we extend the KB with an axiom $\top \sqsubseteq C \sqcup \neg C$ for each concept C s.t. $x : C$ is a concept atom in q. Hence, we obtain a decision procedure for conjunctive query answering, if we can show the following:

Claim 5.1 There is a q-mapping $\sigma_{\mathcal{I}}$ for each model $\mathcal{I}$ of $\mathcal{K}$ iff there is a q-mapping ϕ from q to each completed and clash-free completion graph of $\mathcal{K}$.

To capture the length of the paths in the query, blocking has to be delayed appropriately. In CARIN (i.e., for a logic without transitive roles) this is achieved by using two isomorphic trees (instead of two isomorphic pairs as it is the case for normal pairwise blocking as in $\mathcal{SHIQ}$) s.t. the depth of the trees corresponds to the longest path in the query.

The "if" direction of Claim 5.1 is relatively straight forward but, for the "only if" direction, we have to show (in contrapositive form) that, if there is a completion graph $\mathbf{G}$ for $\mathcal{K}$ s.t. there is no q-mapping ϕ from q onto $\mathbf{G}$, then there is a model $\mathcal{I}$ of $\mathcal{K}$ s.t. there is no q-mapping $\sigma_{\mathcal{I}}$ for $\mathcal{I}$. We now give an example that shows that, even if we find such a completion graph $\mathbf{G}$, we cannot guarantee that there is no mapping $\sigma_{\mathcal{I}}$ for q and the canonical model $\mathcal{I}$ of $\mathbf{G}$ if q contains a transitive role. Let $\mathcal{K}$ be a KB containing the axiom $\top \sqsubseteq \exists r.\top$ for r a transitive and symmetric role and let q be the query $\langle\rangle \leftarrow \langle x, x\rangle : r \wedge \langle x, y\rangle : r \wedge \langle y, y\rangle : r \wedge x : C \wedge y : C$ (Fig. 6, right). The upper part of Fig. 6 shows a possible (simplified) completion graph $\mathbf{G}$ for $\mathcal{K}$, where C or $\neg C$ is added to the node labels to allow for a purely syntactic mapping ϕ. The grey underlying triangle shapes illustrate the two isomorphic trees used for blocking and clearly, there is no q-mapping for $\mathbf{G}$. The lower part of Fig. 6, however, shows that, by unravelling $\mathbf{G}$, we get a structure for which there is a q-mapping. This is so because there is no longer only one node labelled with C, and all role relationships for q are satisfied since r is transitive and symmetric (inferred relationships are not explicitly pictured). Even choosing a larger depth for the two trees would allow this to happen, since the decision for C or $\neg C$ was purely non-deterministic.


r
n1:{¬C} n2:{¬C} n3:{¬C} n4:{¬C} n5:{C} n6:{¬C} n7:{¬C} n8:{¬C}
x:{C} y:{C}
n1:{¬C} n2:{¬C} n3:{¬C} n4:{¬C} n5:{C} n2':{¬C} n3':{¬C} n4':{¬C} n5':{C} n2'':{¬C}
Figure 6: A completion graph $\mathbf{G}$ with trees for blocking (top), a partial unravelling of $\mathbf{G}$ (bottom), and a query graph (right).

Therefore, the absence of a q-mapping for the completion graph does not give us sufficient reasons to show that there is a model for which there is no q-mapping.

We encounter similar difficulties when using binders and variables. It is difficult to determine how "deep" the completion graph has to be before it is safe to block. If the completion graph is not "deep enough", unravelling leads to a clash in the tableau. Since it is, to the best of our knowledge, not known if conjunctive query answering is decidable for $\mathcal{SHIQ}$ or $\mathcal{SHOIQ}$, this might indicate that it is not. We believe, however, that the problem is decidable and that a different proof technique can be used to show this. Our future work is aimed at investigating this.

6 Conclusions

In the previous sections, we have presented two ideas that allow an extension of the rolling-up technique also to cyclic conjunctive queries for $\mathcal{SHQ}$ and $\mathcal{SHOQ}$. For $\mathcal{SHQ}$ we achieved this by extending the logic with a restricted form of binders ($\downarrow$) and state variables as known from Hybrid Logics. This allows us to also express cyclic queries as concepts since binders and variables can be used to express a co-reference. Query entailment can then be reduced to deciding concept satisfiability for the extended DL. However, adding the binder ($\downarrow$) even in the very restricted form needed for query answering has a notable impact on the resulting logic; e.g., for $\mathcal{SHQ}$, this extension leads to the loss of the finite model property. In Section 3.1, we illustrate how a tableaux algorithm for $\mathcal{SHQ}$ can be extended in order to handle query concepts. Although each $\downarrow$ introduces a fresh nominal on the fly, we show that, for $\mathcal{SHQ}$ termination can be regained by ignoring them in the blocking condition. Therefore, we sketched a way of how conjunctive queries with transitive roles in the query body can be answered, which was previously an unsolved problem.

In Section 4, we extend the work by Calvanese et al. [3] to $\mathcal{SHOQ}$, i.e., to a logic that allows for nominals. We do this by proposing a more sophisticated guessing technique, which then again enables the rolling-up of a query into a $\mathcal{SHOQ}$-concept. This suggests that conjunctive query answering for $\mathcal{SHOQ}$ is decidable, which has not been shown before.

In Section 5, we highlight why none of the proposed techniques extends easily to a logic with inverse roles. We also show this for a query entailment algorithm in the style of CARIN [9]. The main problem is to decide when a completion graph is expanded "far enough" to decide if the query is also not entailed in its possibly infinite canonical model.

The rolling-up technique with binders and variables integrates seamlessly into the existing tableaux algorithms, whereas, for the CARIN-style algorithms, the search for a q-mapping is an additional and completely separated step. Further on, the added axioms $\top \sqsubseteq C \sqcup \neg C$ for every concept C in the query, can significantly increase the amount of non-determinism. This makes binders and variables the much more attractive choice from an implementation point of view.

Therefore, our future work will include efforts to show that arbitrary conjunctive queries for $\mathcal{SHIQ}$ and $\mathcal{SHOIQ}$ are decidable. If that is the case, and we believe it is, we aim at extending the rolling-up technique with binders and state variables to $\mathcal{SHIQ}$, $\mathcal{SHOQ}$ and $\mathcal{SHOIQ}$. This would provide a (hopefully practical) decision procedure for arbitrary conjunctive queries over OWL-DL knowledge bases.

References

[1] P. Blackburn and J. Seligman. Hybrid languages. *Journal of Logic, Language and Information*, 4(3):251–272, 1995. Special issue on decompositions of first-order logic.

[2] P. Blackburn and J. Seligman. *Advances in Modal Logic*, volume 1, chapter What are hybrid languages?, pages 41–62. CSLI, 1998.

[3] D. Calvanese, G. De Giacomo, and M. Lenzerini. On the decidability of query containment under constraints. In *Proc. of PODS'98*, pages 149–158. ACM, 1998.

[4] B. Glimm and I. Horrocks. Handling cyclic conjunctive queries. In *Proc. of DL'05*, Edinburgh, UK, July 26–28 2005. CEUR.

[5] I. Horrocks and U. Sattler. Ontology reasoning in the $\mathcal{SHOQ}$ description logic. In *Proc. of IJCAI'01*, pages 199–204. Morgan Kaufmann, 2001.

[6] I. Horrocks, U. Sattler, S. Tessaris, and S. Tobies. How to decide query containment under constraints using a description logic. In *Proc. of LPAR'00*, LNAI. Springer, 2000.

[7] I. Horrocks, U. Sattler, and S. Tobies. Reasoning with Individuals for the Description Logic $\mathcal{SHIQ}$. In *Proc. of CADE'00*, number 1831 in LNAI, pages 482–496. Springer-Verlag, 2000.

[8] U. Hustadt, B. Motik, and U. Sattler. A decomposition rule for decision procedures by resolution-based calculi. In *Proc. of LPAR'04*, volume 3452 of *LNCS*. Springer, 2004.

[9] A. Y. Levy and M.-C. Rousset. CARIN: A representation language combining horn rules and description logics. In *European Conf. on Artificial Intelligence*, pages 323–327, 1996.

[10] M. Marx. Narcissists, stepmothers and spies. In *Proc. of DL'02*, volume 53. CEUR, 2002.

[11] S. Tessaris. *Questions and answers: reasoning and querying in Description Logic.* PhD thesis, University of Manchester, 2001.