The Coherent Multi-Representation Problem with Applications in Structural Biology

Antonio Mucherino

The Coherent Multi-Representation Problem with Applications in Structural Biology

A. Mucherino

IRISA, University of Rennes, Rennes, France.
Email: antonio.mucherino@irisa.fr

Abstract. We introduce the Coherent Multi-representation Problem (CMP), whose solutions allow us to observe simultaneously different geometrical representations for the vertices of a given simple graph. The idea of graph multi-representation extends the common concept of graph embedding, where every vertex can be embedded in a domain that is unique for each of them. In the CMP, the same vertex can instead be represented in multiple ways, and the main aim is to find a general multi-representation where all the involved variables are "coherent" with one another. We prove that the CMP extends a geometrical problem known in the literature as the distance geometry problem, and we show a preliminary computational experiment on a protein-like instance, which is performed with a new Java implementation specifically conceived for graph multi-representations.

1 Introduction

In several applications, given a certain number of constraints involving a given set of objects, the main goal is to identify suitable geometrical representations for such objects. This work takes as a starting point the works in the context of Distance Geometry (DG), where embeddings of a given graph G in a Euclidean space need to be defined in such a way to satisfy a certain number of distance constraints, where the distance metric is generally the Euclidean norm [10].

Among the DG applications, the traditional ones (i.e. the ones that mostly appear in the scientific literature) are the applications in structural biology and sensor networks. Experimental techniques such as Nuclear Magnetic Resonance (NMR) can in fact provide estimates on the proximity between atom pairs in a given molecule, so that looking for molecular conformations satisfying these distance constraints is a problem falling in the context of DG [3, 15]. In sensor networks, the radio signals between two mobile antennas can also provide proximity information, so that localizing the sensors from the estimated distances is basically this very same problem, where the only difference can be found in the quality of the distances that come to play in the definition of the constraints [2, 13]. Other emerging DG applications include acoustic networks [4] and their use in robotics [5], as well as the recent adaptive maps [8].

The motivation for this work comes from the observation that distances are not the only type of information that is given for the definition of the geometric constraints. In structural biology, not only distances are available, but also a certain number of torsion angles formed by quadruplets of atoms in the molecule [11], named *dihedral angles*.

Under certain assumptions, these torsion angles may be converted to distances [7], and the obtained distances may simply be used together with the original distance information. However, conversion procedures cannot guarantee that the performed conversions are lossless. For this reason, we found the need to define a more general problem, where one can deal with several types of information at the same time.

Even though the focus of this work is mostly on the application in structural biology, it is important to point out that the necessity to consider different types of information for the definition of the geometric constraints is also valid for other applications. In sensor networks, in spite of the fact that several works only focus on distances (see for example [9]), there is another type of information that can be measured: the *angle-of-arrival* of the radio signal on the antennas [13].

The main DG problem, known in the literature under the acronym DGP, asks whether it is possible to find an embedding $x : V \to \mathbb{R}^K$ for a simple weighted undirected graph $G = (V, E, d)$ such that the distances between two embedded vertices u and $v \in V$ correspond to the edge weights given by the weight function d. It is generally supposed that G is connected, otherwise the DGP could simply be divided in as many sub-problems as the number of disconnected components in the graph. In some special situations, the properties of G allow us to discretize the search space for potential embeddings x of the graph [14], and this is possible through the introduction of ad-hoc vertex orders on G [6, 18]. In this work, we will suppose that a vertex order on G is available, and that it is given through the orientation of the edges of the graph. In other words, we will suppose that our graphs are directed, and hence denote their arcs with the symbol $(u, v) \in E$.

The Coherent Multi-representation Problem (CMP) is therefore introduced in this work to provide new theoretical basis for the solution of problems arising in the applications mentioned above where different types of information can be employed at the same time. Differently from the DGP, the CMP does not make any net distinction between known (e.g. the distances) and unknown information (e.g. the embedding), but it actually represents the full piece of information in one unique multi-representation system. Thus, a solution to the CMP is a set of values for the internal variables for the several employed representations that turns out to be "coherent" (see Section 2).

This work comes with the initial commit of a new public GitHub repository containing some Java classes implementing the multi-representation system (see Section 3). Throughout the paper, we make reference to types of representations for the vertices that are typical in applications in structural biology, and a very preliminary computational experiment on a protein-like helical model is commented in Section 4. Finally, some future research directions are given in Section 5.

We point out that the idea of employing multi-representations is not completely new in the scientific literature. One important example is given by the works in education and learning [1], where the idea to have complementary representations for the same object, and to simultaneously exploit the advantages that each of them can give, is already explored. In computer science, an example of multi-representation can be found in [20], where some geographic maps at different levels of resolution are managed in one unique database. Some map representations, however, may not be compatible, and they may need to be reused independently from each other. In this work, there is one important aspect that comes to play: we attempt to have the several representations

for a single object to be compatible to one another at all times. To this purpose, we introduce the idea of "coherence" for all involved vertex representations. A similar idea was previously exploited in the context of education in [19].

2 Coherent Multi-representation Problem

Let $G = (V,E)$ be a simple directed graph, where the orientation of its arcs gives us the information about the ordering of the vertices in V. We exploit in this work the fact that each vertex of G can be embedded in the traditional Euclidean space by employing different types of coordinate systems. The most common naturally is the Cartesian coordinate system, but others are also possible. One alternative coordinate system that we will use in the following for our case study in dimension 3 is given by the well-known spherical coordinates, where the location of the vertex $v \in V$ is relative to another vertex u, and it is defined through the distance and the relative orientation of v w.r.t. u in the given Cartesian system. Another representation that we use is through torsion angles [12]. In order to switch from one coordinate system to another, coordinate transformations can be applied.

We use the symbol Y for the definition domain of a given coordinate system. Notice that the dimension of Y may differ from the dimension of the original Euclidean space $\mathbb{R}^K$. An example where these two dimensions are different is given for example by the representation through torsion angles, where the Euclidean space has dimension 3 and the torsion domain has only one dimension. We refer to a coordinate transformation, capable to convert the coordinates from one system to another, as a triplet (P,Y,f), where P is the set of parameters, Y is the coordinate domain, and f is the mapping:

$$f : (p,y) \in P \times Y \longrightarrow x \in \mathbb{R}^K,$$

that performs the transformation from the variables $y \in Y$ to the standard Cartesian coordinates, denoted by x as in the Introduction.

We remark that the trivial transformation (P,Y,f), where $P = \emptyset$, $Y = \mathbb{R}^K$ and f is the identity function, maps Cartesian coordinates into the same Cartesian coordinates. The spherical coordinates require a reference vertex u for a given vertex $v \in V$, so that the Cartesian coordinates of u need, as a consequence, to be included in P. For the definition of torsion angles, the parameter set P contains much more information, which we omit to comment here for lack of space; the interested reader can find all the necessary details in [12]. In the following, we will refer to the triplets (P,Y,f) as transformations, as well as "representations" for a given vertex v.

The definition domains Y can encapsulate geometric constraints, so that the corresponding y variables are only able to cover restricted regions. For example, simple constraints in $\mathbb{R}^K$ can define box-shaped regions. In dimension 3, simple constraints on the distance involved in the definition of the spherical coordinates allow us to control the relative distance between the two involved vertices. Also, simple constraints on the possible values of torsion angles allow us to define arc-shaped regions of the original Euclidean space. For values for y that are not in the delimited region Y, we can consider the existence of a projection operator that projects those values on the allowed coordinate space Y.

One peculiarity in this work consists in associating vertex representations to the arcs of the graph G, and not its vertices. Let $\hat{E}$ be the set obtained as the union of E with the set $\{(v,v) : v \in V\}$; we will use the subscripts $e = (u,v) \in \hat{E}$ to make reference to vertex representations (P_e, Y_e, f_e) that are related to a specific arc of the graph. The predecessor u of v in the arc e is supposed to play the role of reference in the representation of the vertex v. Notice that, for the torsion angle case, there are actually three reference vertices, but it makes sense to associate the torsion angle to the arc (u,v) where u is the farthest reference in the vertex order induced by the orientation of the arcs. Finally, representations which make use of no reference vertices, as it is the case for Cartesian coordinates, can be associated to the added arcs of the type $(v,v) \in \hat{E}$.

For a given arc $e \in \hat{E}$, we can define multiple representations for this same arc; we will use superscripts to distinguish among the various employed representations. The number of representations for the each arc e can vary: there can be only one, or several, or even no one. However, this last situation is generally to be avoided, for some information encoded by the graph would not be exploited in this case. Naturally, all functions f_e involved in the vertex representations need to have a common codomain, which is predetermined and set to $\mathbb{R}^K$ in this work.

Definition 2.1. *Given a vertex $v \in V$ and one of the arcs $e = (u,v) \in \hat{E}$, a "vertex multi-representation" in dimension $K > 0$ and w.r.t the arc e is the set*

$$\Xi_e = \left\{ (P_e^1, Y_e^1, f_e^1), \ldots, (P_e^r, Y_e^r, f_e^r) \right\}$$

containing r different vertex representations related to the arc e. The "expected" Cartesian coordinates for Ξ_e are given by the function:

$$\xi_e : (y_e^1, \ldots, y_e^r) \in Y_e^1 \times \ldots Y_e^r \longrightarrow \frac{1}{r} \sum_{i=1}^{r} f_e^i(p_e^i, y_e^i) \in \mathbb{R}^K, \tag{1}$$

where $p_e^i \in P_e^i$, for each i. We say that the set of internal coordinates $y_e = (y_e^1, \ldots, y_e^r)$ is "coherent" if

$$\forall i \in \{1, \ldots, r\}, \quad \xi_e(y_e) = f_e^i(p_e^i, y_e^i).$$

Vertex multi-representations allow us to simultaneously associate several types of representations to the same vertex v of the graph. For example, the use of spherical coordinates can help controling the distance between v and its reference u, while v can at the same time also be represented through a torsion angle. Recall that the domain Y associated to the transformations makes it possible to restrict the region of feasibility for the variables. In some special cases, the domain Y may allow some variables to take only value, thus indicating that the value for this variable is actually known (or imposed).

Def. 2.1 can be trivially extended to all the arcs in the edge set $\hat{E}$, so that a multi-representation for all the vertices $v \in V$, and for all arcs $e = (u,v) \in \hat{E}$, can be defined. When we extend to the entire graph G, we can notice that there are actually two levels of multiplicity for the representations. Firstly, for a given arc $(u,v) \in \hat{E}$, several different representations can be defined, as commented a few lines above. Moreover, the second level of multiplicity comes from the fact that at least one representation is expected to be defined for every arc in $\hat{E}$, and hence there will be at least as many representations for v as there are arcs $(u,v) \in \hat{E}$ having v as a destination vertex.

When all arcs in $\hat{E}$ are involved, we say that the set of vertex multi-representations $\Xi_G = \cup_{e\in\hat{E}}\Xi_e$ forms a "graph multi-representation". The function in equ. (1) can also be extended to the entire graph, and we will refer to it with the symbol ξ_G. The input of ξ_G is a vector $\bar{y}$ combining all the variables y_e^i, for each arc $e \in \hat{E}$ and each representation (indexed by i). We say therefore that $\bar{y}$ is *coherent* if all its components are coherent (see Def. 2.1).

The only possibility for the function ξ_G to be equivalent to a standard graph embedding is when the graph is trivial. This situation is naturally of no interest, and we will therefore suppose that our graphs G are non-trivial. The graph multi-representation evidently allows for a richer representation of the graph, which explicitly exploits the connectivity information encoded by the graph, which is instead not taken into consideration by standard graph embeddings.

Definition 2.2. *Given a simple directed graph $G = (V,E)$ and a graph multi-representation Ξ_G in dimension $K > 0$, the Coherent Multi-representation Problem (CMP) asks whether there exists a vector $\bar{y}$, composed by the internal variables for the coordinates of the vertex multi-representations in Ξ_G, that is coherent.*

The following result relates the DGP (see Introduction) with the new CMP. An immediate consequence of this result is that the CMP is NP-hard.

Proposition 2.3. *The DGP is a special case of the CMP.*

Proof. Let $G^{dgp} = (V,E,d)$ be a simple weighted undirected graph representing a generic instance of the DGP. Let $G = (V,E)$ be the same graph G^{dgp} without the weight function d, but with directed edges for encoding a suitable vertex order (see Introduction). If no information about vertex orders on V is available, any order can be used.

We will proceed by constructing a graph multi-representation Ξ_G. For every $v \in V$, we assign a representation by Cartesian coordinates to each arc of the type $(v,v) \in \hat{E}$. For every $(u,v) \in E$, moreover, we assign a representation by spherical coordinates, where the domain Y encodes the bounds on the distance values given by the weights of the graph G^{dgp}. Notice that more than one distance may be known for a given vertex v, and hence the number of introduced representations by spherical coordinates is likely to vary vertex per vertex.

By definition, a solution to the so-constructed CMP is a vector $\bar{y}$ containing the internal coordinates for all these vertex representations (the introduced Cartesian and spherical coordinates). The vector $\bar{y}$ is supposed to be coherent, in the sense that, for every $e \in \hat{E}$, all functions f_e^i are supposed to give the same result. As a consequence, the expected Cartesian coordinates for the generic vertex v must satisfy all distance constraints encoded by the spherical coordinates, and they form therefore a solution to the original DGP. □

3 An object-oriented implementation

A quick search on the Internet can reveal the existence of several freely distributed implementations for storing graph structures, in several low and high level programming

languages. For this work, we opt for a completely new implementation which does not only allow us to store the graph structure, but also the several vertex representations that we can associate to its arcs. Moreover, our implementation exploits an internal notification system for keeping all vertex representations updated. Naturally, it is not always possible to keep all vertex representations in a coherent state. We have implemented specific methods for this verification.

The notification system acts at two levels: it makes sure that all representations for the same vertex are synchronized (as far as this is possible), and it notifies other vertex representations, that use the modified vertex as a reference, of the performed change. These other representations will have to update their internal state in order to consider the new information provided by their references.

The language we have chosen is Java for its object-oriented paradigm and for its relative simplicity; specific syntax related to more recent versions of Java have been avoided, so that translations to other languages supporting classes, encapsulation and inheritance will potentially be easy to perform. The Java codes are available on a public GitHub repository[1]. The current implementation focuses on coordinate systems in a three-dimensional Euclidean space only.

The main class in our Java code is the `Coordinates` class. As its name suggests, this class is supposed to hold information, and to perform actions on, the numerical values employed for the various vertex representations. The peculiarity of this class is that it contains a certain number of private sub-classes, each defining a particular representation and some basic methods for their manipulation. We point out that here we relax the encapsulation principle for the access to the attributes of the sub-classes. We found in fact that a stricter encapsulation would have led to a more complex and less efficient code without really adding any level of access security to the data. Notice however that the encapsulation principle is respected at the level of the `Coordinates` class. This class, moreover, contains a certain number of private methods that are not supposed to be invoked from the outside, which ensure the realization of the notification system mentioned above.

Every instance of `Coordinates` can hold several representation types, and for every type, it can hold several instances for the given type. The only exception is for the standard Cartesian coordinates, because two instances of the Cartesian coordinates that are coherent in the same `Coordinates` object can only be identical. This is however not true for the spherical coordinates, and for the torsion angles.

Suppose for example that an instance of `Coordinates`, call it `C1`, contains three inner representations: one of Cartesian type, one of spherical type, and another of torsion type. Suppose we wish to position `C1` in specific Cartesian coordinates. Our class provides a method to this purpose, but it does not only limit itself to change the values of the Cartesian coordinates, it also attempts to adapt all other representations to the new imposed Cartesian coordinates. In practice, the variables used for the spherical and torsion types are updated for keeping them compatible with the new Cartesian coordinates. However, as mentioned above, this is not always possible, because the definition domain of the internal variables may be constrained (through the set Y introduced in

[1] `https://github.com/mucherino/DistanceGeometry`, commit `be4e33b`, folder `javaCMP`.

Section 2, see the `BoundedDouble` class). When the update fails, then we say that `C1` is not in a coherent state.

Now suppose for simplicity that the current state of `C1` is coherent. Suppose that the reference `Coordinates` instance for `C1`, in both spherical and torsion types, is `C0`. `C0` admits only one representation, whose type is Cartesian. When the Cartesian coordinates of `C0` are modified, our notification system sends a "signal" to `C1`, which is now supposed to update some of its internal variables. First of all, we can recompute the Cartesian coordinates of `C1`, by using the information given by the spherical type. However, `C0` takes also part in the definition of the torsion angle, which may now be incompatible with the spherical type. All possible updates, for all types in `C1`, are attempted, but, again, the procedure may fail in leaving `C1` in a coherent state. The little experiment commented in the next section was instead conceived in such a way to maintain the coherent state for its representations.

The `Coordinates` class contains methods for verifying whether an instance is in a coherent state or not, and it can provide, in both situations, its "expected" Cartesian coordinates via a devoted Java method implementing the formula in Def. 2.1. The interested reader is invited to look directly at the code for additional technical details of this implementation.

4 A simple experiment

We present in this section a very preliminary experiment performed with our new Java classes. The details of the experiment can be found on the GitHub repository in the class named `Experiments` (please make sure to refer to the commit indicated in the previous section). Our experiment makes use of the torsion type, which is particularly useful in the construction of the typical helical structures in protein conformations. The reader can find other smaller experiments, involving the spherical type as well, in the set of automatic tests implemented in the `main` method of the `Coordinates` class.

In `Experiments`, we initially construct our helical conformation by simply instantiating a multi-representation for the underlying graph where standard Cartesian coordinates and torsion angles co-exist. We do not give values to the Cartesian coordinates, we basically only provide the torsion angles, and this action automatically gives us the model depicted at the bottom of Fig. 1. The interesting point about the experiment is that the user is only supposed to initialize and set up the values of the torsion angles, whereas the calculations that allow the construction of the model are automatically performed by our Java code in an attempt to keep the various representations coherent with one another.

The second model, at the top of Fig. 1, was obtained with one line of code in the `Experiments` class, consisting in changing the value for one torsion angle. In this case, not only the change in one representation type started the updating system within the same `Coordinates` instance (the one whose torsion angle was changed), but it also notified, in a chain, all *subsequent* `Coordinates` instances, which updated their inner variables as well.

This automatic synchronization of the various representations can be particularly useful when implementing solution methods that mainly act on specific coordinate sys-

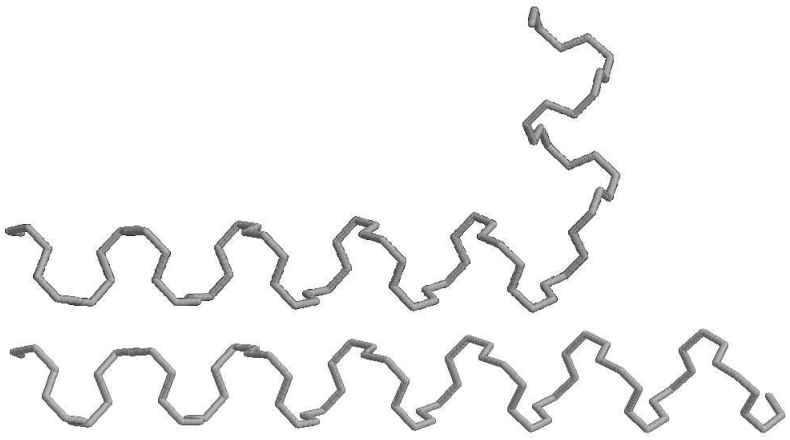

Fig. 1. Two helix-like confirmations obtained with our new Java classes.

tems, because it allows for simultaneously controling other vertex representations, by immediately revealing, for example, that some geometric constraints are consequently violated.

5 Conclusions

This work introduced the CMP, a new decision problem that extends another very common and widely studied problem, named the DGP. We showed that the DGP is included in the CMP, so that the latter inherits the NP-hardness of the former. We have briefly presented a new Java implementation allowing for vertex multi-representations that are at the core of the CMP.

Future works will be mainly devoted to the development of solvers for the CMP, by taking as a starting point some previous works that we have performed in the context of the DGP (see for example [16]). Moreover, it is also our intention to include other vertex representations in our implementations, in order to tackle a larger variety of applications. The realization of this future work is likely to be particularly delicate, for it may reveal some limitations of our multi-representation approach. In fact, some types of information may not be capable to uniquely reconstruct the Cartesian coordinates of the vertices, or at least not with the same precision as the spherical and torsion types can do. Finally, we also plan to study, in a near future, the use of the CMP in the context of dynamical problems [17].

Acknowledgments

We wish to thank the three reviewers for their fruitful comments. This work is partially supported by ANR French funding agency (MULTIBIOSTRUCT project ANR-19-CE45-0019).

References

1. S. Ainsworth, *DeFT: A Conceptual Framework for Considering Learning with Multiple Representations*, Learning and Instruction **16**(3), 183–198, 2006.
2. P. Biswas, T. Lian, T. Wang, Y. Ye, *Semidefinite Programming based Algorithms for Sensor Network Localization*, ACM Transactions in Sensor Networks **2**(2), 188–220, 2006.
3. G.M. Crippen, T.F. Havel, *Distance Geometry and Molecular Conformation*, John Wiley & Sons, 1988.
4. I. Dokmanić, R. Parhizkar, A. Walther, Y.M. Lu, M. Vetterli, *Acoustic Echoes Reveal Room Shape*, Proceedings of the National Academy of Sciences **110**(30), 12186–12191, 2013.
5. F. Dümbgen, A. Hoffet, M. Kolundzija, A. Scholefield, M. Vetterli, *Blind as a Bat: Audible Echolocation on Small Robots*, IEEE Robotics and Automation Letters **8**(3), 1271–1278, 2023.
6. D.S. Gonçalves, A. Mucherino, *Optimal Partial Discretization Orders for Discretizable Distance Geometry*, International Transactions in Operational Research **23**(5), 947–967, 2016.
7. S.B. Hengeveld, T. Malliavin, L. Liberti, A. Mucherino, *Collecting Data for Generating Distance Geometry Graphs for Protein Structure Determination*, Proceedings of ROADEF23, Rennes, France, 2 pages, February 2023.
8. S.B. Hengeveld, F. Plastria, A. Mucherino, D.A. Pelta, *A Linear Program for Points of Interest Relocation in Adaptive Maps*, to appear in Lecture Notes in Computer Science, Proceedings of Geometric Science of Information (GSI23), St. Malo, France, August 2023.
9. N. Krislock, H. Wolkowicz, *Explicit Sensor Network Localization using Semidefinite Representations and Facial Reductions*, SIAM Journal on Optimization **20**, 2679–2708, 2010.
10. L. Liberti, C. Lavor, N. Maculan, A. Mucherino, *Euclidean Distance Geometry and Applications*, SIAM Review **56**(1), 3–69, 2014.
11. T.E. Malliavin, A. Mucherino, C. Lavor, L. Liberti, *Systematic Exploration of Protein Conformational Space using a Distance Geometry Approach*, Journal of Chemical Information and Modeling **59**(10), 4486–4503, 2019.
12. T.E. Malliavin, A. Mucherino, M. Nilges, *Distance Geometry in Structural Biology: New Perspectives*. In: [15], Springer, 329–350, 2013.
13. G. Mao, B. Fidan, B.D. Anderson, *Wireless Sensor Network Localization Techniques*, Computer Networks **51**(10) 2529–2553, 2007.
14. A. Mucherino, C. Lavor, L. Liberti, *The Discretizable Distance Geometry Problem*, Optimization Letters **6**(8), 1671–1686, 2012.
15. A. Mucherino, C. Lavor, L. Liberti, N. Maculan (Eds.), *Distance Geometry: Theory, Methods and Applications*, 410 pages, Springer, 2013.
16. A. Mucherino, J-H. Lin, D.S. Gonçalves, *A Coarse-Grained Representation for Discretizable Distance Geometry with Interval Data*, Lecture Notes in Computer Science **11465**, Lecture Notes in Bioinformatics series, I. Rojas et al (Eds.), Proceedings of the 7^{th} International Work-Conference on Bioinformatics and Biomedical Engineering (IWBBIO19), Part I, Granada, Spain, 3–13, 2019.
17. A. Mucherino, J. Omer, L. Hoyet, P. Robuffo Giordano, F. Multon, *An Application-based Characterization of Dynamical Distance Geometry Problems*, Optimization Letters **14**(2), 493–507, 2020.
18. J. Omer, A. Mucherino, *The Referenced Vertex Ordering Problem: Theory, Applications and Solution Methods*, Open Journal of Mathematical Optimization **2**, article no. 6, 29 pages, 2021.
19. T. Seufert, *Supporting Coherence Formation in Learning from Multiple Representations*, Learning and Instruction **13**(2), 227–237, 2003.

20. S. Zhou, C.B. Jones, *A Multi-representation Spatial Data Model*. In: “Advances in Spatial and Temporal Databases”, conference proceedings of the 8^{th} International Symposium SSTD03, Santorini Island, Greece, 394–411, Springer, 2003.