**Simplified Spatio-Temporal Data Model for CHGIS**

Merrick Lex Berman
China Historical GIS
Harvard Yenching Institute
November 2002

**Abstract:**

In the compilation of datasets for the CHGIS spatio-temporal database we must ask make sure that we have adequately answered the following questions:

(1) can we create a slice in time for a particular date?
(2) can we search for placenames and find the correct record among ambiguous results?
(3) can we sort by administrative feature types?
(4) can we determine the place in the administrative hierarchy for a particular place?
(5) can we show how a particular place changes over time?

The following outline of the CHGIS database design examines the ways in which the current data model satisfies these basic requirements, with illustrated examples of the basic problems encountered when trying to keep track of spatial objects as they and their attributes changes over time.

### 1.1 Rules Regarding Creation of Records in the Database

According to the basic requirements for the CHGIS project, there were three primary types of change to be recorded in the database:

1) change in placename
2) change in administrative status
3) change in location, area, or boundary

This proves to be a sound basis for tracking historical places, as it has also emerged as the basis for tracking places in historical gazetteer projects, such as the Alexandria Digital Gazetteer Content Standard, which defines the core elements needed for any digital gazetteer entry for an historical place as: *placename, category, footprint*. These are directly comparable with our *placename, administrative type,* and *spatial object*.

Therefore the CHGIS model includes a basic table, called the historical instance table, to which a new row is added whenever one of the three types of changes occurs. For our purposes an *historical instance* is defined as a record describing a span of time during which the *placename*, *administrative type*, and *spatial object* all remained unchanged, lasting from a specific begin date to a specific end date. Should any of these three attributes change, a new record is added to the database. In the example below, we show records for two counties, A and B, and several towns, 1, 2, 3. No historical changes are shown here, except an indication that the a change occurred affecting county A at the end date: time 2.

**historical instance table**

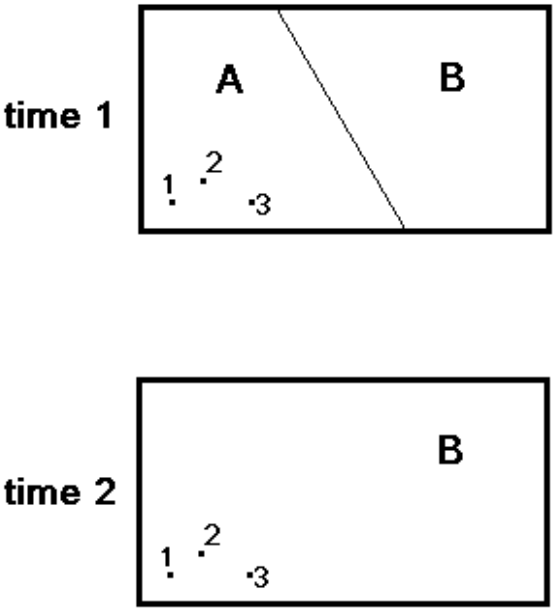
| name | type | begin | end |
|---|---|---|---|
| county A | county | time 1 | time 2 |
| county B | county | time 1 | |
| town 1 | town | time 1 | |
| town 2 | town | time 1 | |
| town 3 | town | time 1 | |

### 1.2 Two Elements of Administrative Status

One thing that we must further clarify is the definition of "administrative status," which is composed of two quite different elements. First, we have *administrative type*, a kind of feature type, which is simply the category of a particular administrative unit that applies to a particular record. With *administrative type* we are simply defining the type of unit for a record, such as "county," or "town," as shown in the preceding table. The second element of "administrative status" is the *part of relationship* that is needed to understand the place in the administrative hierarchy. For example, in the table above, we know that 1, 2, and 3 are "towns," but we don't know whether they are under the jurisdiction of county A or county B.

To keep track of the administrative hierarchy, we need to establish relationships between records in the historical instances table showing "containership," which is to say that one record is *part of* another record. An example would be Longling County is *part of* Baoshan District, and Baoshan District is *part of* Yunnan Province. In this usage *part of* implies both administrative jurisdiction and physical area, so that all of the area under the jurisdiction of Longling county must be contained within the area of Baoshan District.

An example will illustrate the clear difference between an *administrative type* and the *part-of relationship* between different administrative units in the database. Consider the following illustration:




Imagine at time 1, that county A and county B are *part of* prefecture X. County A has its own subordinate units, towns 1, 2, 3. Then a change occurs: at time 2, county A is abolished and all of its territory is transferred to the jurisdiction of county B. So the database must track not only county A's dissolution, but also the change of relationship for towns 1, 2, 3, which became *part of* county B.

### 1.3 Tracking the Administrative Hierarchy with Part-Of Table

Regardless of whether they are being examined at time 1 or time 2, the subordinate towns 1, 2, 3 do not change and maintain the same *adminstrative type* = town. By contrast, beginning from time 1 and ending at time 2 they have a *part of* relationship as subordinate to county A, whereas beginning at time 2 that relationship ENDS, while a new *part of* relationship, being subordinate to county B, BEGINS. Therefore the most expedient way to handle this information is to simply enter each unique *part of relationship* as a new record in a separate part-of table and to make sure the END date of the *part of* record is correct for the first relationship, and to add a new record for the second *part of* relationship.

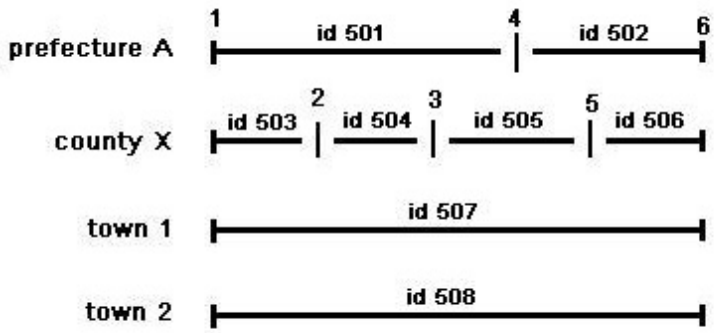
**historical instance table**

| name | type | begin | end |
|---|---|---|---|
| county A | county | time 1 | time 2 |
| county B | county | time 1 | |
| town 1 | town | time 1 | |
| town 2 | town | time 1 | |
| town 3 | town | time 1 | |

**part-of table**

| name | part-of | begin | end |
|---|---|---|---|
| town 1 | county A | time 1 | time 2 |
| town 2 | county A | time 1 | time 2 |
| town 3 | county A | time 1 | time 2 |
| town 1 | county B | time 2 | |
| town 2 | county B | time 2 | |
| town 3 | county B | time 2 | |

By keeping track of relationships in a separate part-of table we can maintain the original logic of only creating a new record for each change in the historical instance table when the *placename* changes, or the *adminstrative type* changes, or the *spatial object* changes. There is another, more important reason why we do not want changes in *part of* relationships to create new records in the historical instance table: which is to reduce the impact of changes in *part of* relationships to only one level in the administrative hierarchy. Consider the following illustration:




Here we see that prefecture A remains constant in terms of *placename*, *type*, and *spatial object* until time 4, when a change occurs. In our database the before and after incarnations of prefecture A are recorded as rows in the historical instance table, with unique id numbers: **id 501** and **id 502**. Similarly county X, which is subordinate to prefecture A, has four records of historical instances: **id 503**, **id 504**, **id 505**, and **id 506**. Note that the towns subordinate to county X don't have any recordable changes at all for this entire period, and need only one record each: **id 507**, and **id 508**. Now, according to the model suggested above, we would use a separate part-of table to show that the sequential instances of county X were *part of* the instances of prefecture A. To eliminate ambiguity of placenames we must use the unique id numbers for each historical instance record as follows:

| name | id | part-of | part-of-id | begin | end |
|---|---|---|---|---|---|
| county X | 503 | prefecture A | 501 | time 1 | time 2 |
| county X | 504 | prefecture A | 501 | time 2 | time 3 |
| county X | 505 | prefecture A | 501 | time 3 | time 4 |
| county X | 505 | prefecture A | 502 | time 4 | time 5 |
| county X | 506 | prefecture A | 502 | time 5 | time 6 |
| town 1 | 507 | county X | 503 | time 1 | time 2 |
| town 1 | 507 | county X | 504 | time 2 | time 3 |
| town 1 | 507 | county X | 505 | time 3 | time 5 |
| town 1 | 507 | county X | 506 | time 5 | time 6 |
| town 2 | 508 | county X | 503 | time 1 | time 2 |
| town 2 | 508 | county X | 504 | time 2 | time 3 |
| town 2 | 508 | county X | 505 | time 3 | time 5 |
| town 2 | 508 | county X | 506 | time 5 | time 6 |

Consider that there are five *part of* relationship records for county X, because even though county X itself did not change between time 3 and time 5 (**id 505**), its parent prefecture did change at time 4. Therefore, in the historical instance table, we only need one instance for **id 505**, but in the part-of table we need two relationship records to show that county X was *part of* **id 501** until time 4, when it switched to being *part of* **id 502**. The important thing to consider is that the change in relationship of county X to its parent does NOT IMPACT the relationship of subordinate towns to county X! Because county X remained constant from time 3 to time 5 as **id 505**, there is no need to update the *part of* relationship records for town 1 and town 2, which were part of **id 505**. This clearly demonstrates that a change of any parent unit only impacts its children units one level down the hierarchy.

Without going into excessive detail, we should point out that if a single column in the historical instance table was used to indicate the *part of* relationship, it would require splitting child records every time a parent unit changed. In addition, this fragmentation would not be limited to one level down, but would impact ALL subordinate units downwards in the hierarchy, and would result in massive creation of records in the main historical instance table. The records created because of changes in parent would only function as indicators of *part of* relationships, not new historical instances. To make matters worse, there are tens of thousands of smallest level units (such as towns and villages) to keep track of, therefore we came to the obvious conclusion that the optimal database design would limit the number of changes needed for the lowest level units to the greatest extent possible. This is economically achieved by tracking the relationships among historical instances table records in a separate part of table, as described above.

### 1.4 Tracking Sequence of Changes over Time

Having discussed the methodology for keeping track of relationships in the administrative hierarchy, we must also consider relationships across time. We can picture both of these relationships as network systems, which function independently but share the same constituents. The administrative hierarchy is a network beginning with the central political authority of the state, beneath which are subordinate administrative units that have their respective jurisdictions and subordinate units.

Qing Dynasty
- Baoshan Fu
  - Longling Xian
  - Shidian Xian
- Pu'er Fu
  - Simao Ting

Similarly, the temporal sequence is a network of preceding historical instances to subsequent instances.

Mengmao Si
- Ruili Xian
  - Ruili Shi

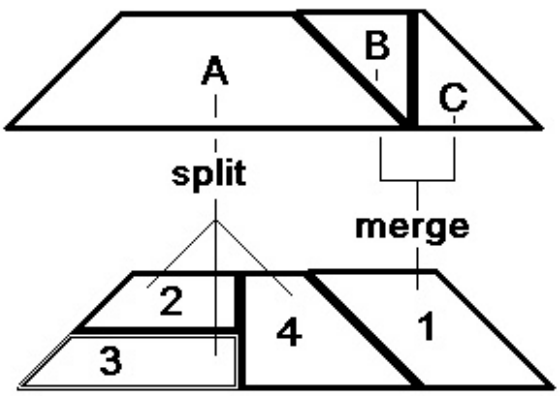
In the temporal network we simply follow the changes recorded in the historical instance table, regardless of whether they are changes in placename, type, or spatial object. We only seek to establish which instance was *preceded by* another instance.

There are three compelling reasons to compile a separate table to keep track of the *temporal sequence* of records in the historical instance table:

1) First, it will allow the compilers and users of the database to easily navigate the succession of records in the database as they need to introduce new records in between two existing instances based on new information.
2) Second, a temporal sequence table will allow the compilers to accurately update the *part of* relationships described above.

3) Third, the ability to track change of any given place over time was one of the original objectives for the CHGIS project, and the only way to actually perform this type of spatio-temporal query is to track the sequence of changes.

Fortunately, tracking the temporal sequence of historical instances is much easier than tracking *part of* relationships. But like *part of* relationships, it must be done in a separate table. There are two basic reasons that creating a separate temporal sequence table is unavoidable—because multiple areal units can merge into a single areal unit, and because records identified as point objects may change location to a different point. First, let's consider a merge, which mandates a one-to-many relationship between the record after a merge and the multiple records that preceded the merge:




In the preceding graphic, we can see that area 1 was preceded by BOTH area B and area C. This would be best handled by two rows in a separate table rather than multiple fields in the historical instance table. In the tables below, historical instances are shown on the left, and *preceded by* relationships are shown in the temporal sequence table on the right:

**historical instance table**

| name | id |
|---|---|
| A | 301 |
| B | 302 |
| C | 303 |
| 1 | 304 |
| 2 | 305 |
| 3 | 306 |
| 4 | 307 |

**temporal sequence table**

| name | id | preceded-by | preceded-id |
|---|---|---|---|
| 1 | 304 | B | 302 |
| 1 | 304 | C | 303 |
| 2 | 305 | A | 301 |
| 3 | 306 | A | 301 |
| 4 | 307 | A | 301 |

Note that we do not need to worry about dates in the temporal sequence table, because they are kept track of in the main historical instance table, and all we need to be able to do is to explicitly track the sequence of changes from one instance to another. In terms of preceding instances merging or splitting into other instances, it doesn't matter whether only a percentage of a an areal unit was merged or split into its subsequent unit, because all we need to do is to IDENTIFY a preceding or subsequent unit. Once we have accomplished that we can simply overlay the spatial objects and see which portion of the antecedent was absorbed, or we can use various GIS utilities to calculate the exact areas involved. The main problem that must be addressed is that unless we keep track of preceding units in the temporal sequence table, we currently have NO WAY to determine which units preceded or followed one another, and will have failed to meet one of our primary objectives.

### 1.5 What Constitutes a "Place" Changing Over Time?

Although we have established the need to keep track of *temporal sequence* in the database, we do foresee several problems in declaring *preceded by* relationships. One problem has to do less with the nature of the records being added to the database then the conception of the term "place." The question is: how do we distinguish between a "place" that has changed from one instance to another, and a completely new "place?" For example, the argument can be made that if a place changes both it's *name* and *administrative type* simulataneously it is no longer the same place, that it has been abolished, and that a new place has been established in its stead.

For example, Fort Dearborn was established at the same location on the shores of Lake Michigan where the city of Chicago was subsequently founded. Are the historical instances of (1) **placename=Fort Dearborn**, **type=military fort**, and (2) **placename=Chicago**, **type=city** really the same place? Or are they two different places? For the purposes of our CHGIS database model, we deliberately seek to forestall this argument by establishing links between all possible sequential instances of places. So we would say, yes, a record must be added to indicate that the historical instance of Chicago was *preceded by* Fort Dearborn. We do this so that the the compilers of the database, who are working directly with extensive historical documentation, will include explicit temporal links between records, allowing the users to discover the pattern of change over time by querying on the *preceded by* column for any particular record.

We understand that there is an inherent artificiality about tracking temporal sequence in this way, as what it accomplishes is to link instances of change rather than clearly defining a "place." However, we have never been able to find a consistent means of differentiating the

transformation of a "place" from the abolition, creation, merging, splitting, or re-establishment of a place. Therefore our solution is to include all of these options as links in a temporal sequence, so that we are at least not eliminating factors that can later be used to reconstruct the change of places over time.

In the previous example, of Fort Dearborn and Chicago, we assumed that the location (and presumably the spatial object) of these two places were one and the same. But it reality we may have one point object for the location of Fort Dearborn (the center of the actual buiding), and a different point object for the location of Chicago (usually based on the location of the City Hall). This is an important thing to keep in mind, because in GIS, there is no reason to assume any point object has any particular relationship with any other point object. Nonetheless, to the historical geographer, the fact of Fort Dearborn's existence as a temporal predecessor to the city of Chicago is absolutely essential. Indeed it doesn't matter if Fort Dearborn continued to exist long after the city of Chicago was founded, in other words, if the *preceded by* instance has a *temporal overlap* with its successor. Since the temporal sequence table itself has no date values, the only job it has to perform is to indicate that some particular instance is deemed to be some kind of predecessor to another instance, and the only absolute rule is that the begin date of the predecessor must fall before the begin date of the successor. In this way, we avoid the issue of *temporal fragmentation*, which would parallel the problem described earlier for *part of* relationships.

### 1.6 Bare Minimum for Spatio-Temporal Data

In the preceding sections we have laid out the basic requirements for the spatio-temporal data model used for CHGIS. To accommodate the historical information available to us, we have settled on a row-versioned database showing each instance of placename, type, and spatial object change, the historical instance table. The administrative hierarchy is derivable from the relationships between instances that are recorded in the part of table. And the change over time of particular places can be obtained by querying the temporal sequence table.

Several other valuable tables, which can be developed later on from the basic three tables, would include an administrative seat table and a change type table.

The former table would be used to indicate the location of the adminstrative seat, or capitol, for a particular historical instance. Because the location of an administrative seat might change at a date that falls in between the begin and end dates of its associated historical instance, the

administrative seat table must also be created separately with its own begin and end dates, exactly duplicating the structure and functionality of the part of table.

The change type table can be constructed once all the possible changes from one instance to another are known. For example, using the basic three tables we can devise a query to find all of the historical instances of **type=fu** which were immediately preceded by **type=xian** within a certain period of time, and within a certain provincial jurisdiction. This would find all of the places at which a *fu*, or prefectural office, was either newly *established*, or *moved* from a different location. A change type table can be developed to clarify these events, for example one change type would be *established*, and another change type would be *moved.* The change type table need not include any date information, but it must allow for more than one change type to be associated with any particular instance—for example, a place may change it's *name* simultaneously with it's *administrative type*. In this way the change type table would be an exact duplicate in structure and functionality with the temporal sequence table.

Many other helpful tables could be devised to aid spatio-temporal queries, but the bare minimum can be accomplished with the three main tables and the following rules for record creation:

A new record must be created in the historical instance table whenever there is a:

1) change in placename
2) change in administrative status
3) change in location, area, or boundary

As changes occur in historical instances, records must be created in the part of table:

1) when a parent jurisdiction changes (but not boundary), the previous instance of *part of* relation must have the end date changed to the match the end date of its original parent, and a new instance must be added to indicate the new parent.
2) when a boundary changes an overlay procedure must be done to check possible impacts on shared boundaries with parents, children or adjacent boundaries. Actual changes in spatial object require a new record to be created in the main historical instance table, and an evaluation to see if there are children contained within the previous footprint which need new *part of* records.

As new records are added to the historical instance table, each of their preceding instances should be indicated by a new row in the temporal sequence table.