RADAR

► Evolution of Earth Observation

Radio Frequency Identification (RFID)

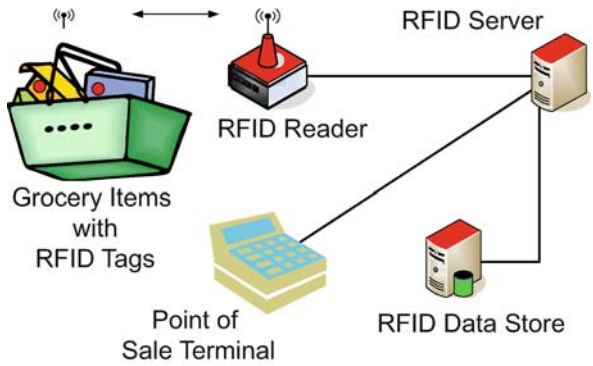
MICHAEL GOSHEY
Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA

Synonyms

RFID; RF identification; Electronic identification; Radio tagging; Electromagnetic tagging

Definition

Radio Frequency Identification (RFID) exists as a distinct subset of the larger family of automated identification technologies that includes things like bar codes, visual scanning devices and biometric readers. RFID is a means of automated identification that features electronic tags used both to store data and to act as transponders for sending the stored data as the payload in electromagnetic waves (radio waves) sent to detached listening devices (RFID readers) [1,2]. The tags can be affixed to animate or inanimate objects by a variety of methods and the readers that receive emissions from the tags translate the wave-embedded data into meaningful information (Fig. 1). They are a significant improvement over bar codes, for example, in that they do not require any human intervention. Currently deployed RFID systems provide real-time identity tracking and monitoring and make possible a wide variety of access control and inventory management solutions [3,4,5]. Beyond this primary value of real-time identification monitoring, the back-end of a typical RFID solution offers a distinct secondary value through storing the rich set of identity data captured by the system, enabling sophisticated data mining techniques to be applied to a number of interesting scenarios [6]. Additionally, while RFID systems primarily deal with automating identity management, they are frequently deployed in combination with Geographic Information Systems (GIS) to yield even more powerful solutions that track both identity and spatial location [7,8].



Radio Frequency Identification (RFID), Figure 1 Simple system diagram of grocery store point of sale RFID

Historical Background

While the practical application of Radio Frequency Identification has grown tremendously in recent years, the technology has a long and interesting history.

1920's–1940's

The seminal paper in the field is considered by many to be *Communication by Means of Reflected Power* by Harry Stockman in 1948 [9]. This paper proposed the key RFID notion: that power generated at the base (often fixed) end of a point-to-point electronic communication pair is reflected and reused in order to power the return transmission from the remote (often mobile) end of that pair. Stockman's paper echoed work that was already being done in *radar* to track the location of an object through the use of reflected radio waves which began in the early 1920's [1]. The 1930's and 1940's saw steady development in both the radio and radar fields, paving the way for Stockman's work, which was also reflected in the progress of another

prominent technology of the era- Identification Friend or Foe (IFF) systems [10]. IFF systems employed transponders in attempts to avoid 'friendly fire' situations and were early predecessors of today's air traffic control systems. They were utilized during World War II to gain military advantage by identifying the allegiance of approaching aircraft before visual confirmation was possible [11,1].

1950's–Present

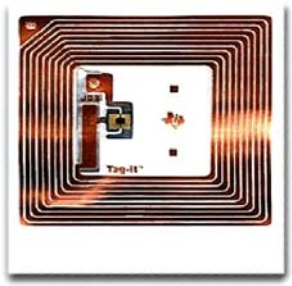
The 1950's saw the first work combining microwaves with RFID [12]. In the 1960's and 1970's the field gradually advanced on multiple fronts: academic research, patent activity and commercialization [1]. There were a number of patents related to remote measurement, communication and activation using radio frequency power, perhaps the most well-known of which was Charles Walton's 1973 patent for keyless door entry using passive RFID [13]. This was an especially important year for RFID as it also saw the first patent dealing with active RFID systems where tag memory could be rewritten and updated [14]. A couple of years later a landmark paper was authored by a group at Los Alamos [15], where work was being conducted on animal tracking and automated vehicle control systems. Large companies such as Phillips, Westinghouse, Raytheon and General Electric were ramping up work on shipping, transportation and factory automation systems based on RFID-related technologies. Automated toll-collection systems were in place both in the United States and in Europe by the mid-1980's and ultra-high frequency (UHF) RFID was developed in the early 1990's by IBM [14]. A team from MIT sponsored by Gillette and Proctor & Gamble later carried UHF-based RFID further, applying it to low-cost supply chain management applications that featured dumb tags containing only identification numbers rather than rewritable memory. Finally, in recent years it has been mega-merchant Wal-Mart (along with fellow retailers Target and Tesco and the U.S. Department of Defense) that has had perhaps the most influential hand in pushing RFID to the forefront of the global technology landscape [5]. Recent mandates by these large organizations to convert all of their vendors and suppliers to RFID-based supply chain management systems have impacted entire industries while raising significant concerns over the potential intrusiveness and other privacy drawbacks now associated with RFID [16,17].

Scientific Fundamentals

A good first step toward understanding a Radio Frequency Identification system is to inspect its primary components. RFID systems are supported through a network of application software, middleware and computing resources. While these play a role in the overall delivery of RFID and are perhaps specialized for this purpose they are not unique to the RFID domain. However at the center of these general actors are several RFID-specific components: tags, readers and label printers [33,34].

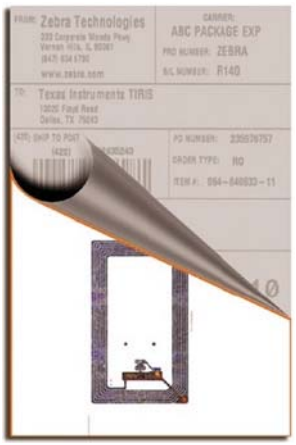
Tags

RFID systems exist for the purpose of capturing information about entities of interest (i. e. things one wishes to identify, track and monitor). In an RFID environment this information is stored on electronic tags. An RFID tag is simply a silicon microchip bundled with an antenna and attached to an entity of interest (Fig. 2). Not surprisingly the chips store identifying data such as unique serial numbers and other information required by the ID system. The tags act as transponders, receiving and responding to radio signals to and from RFID readers. RFID has some distinct advantages over bar code technology, including the ability of tags to function without line of sight and the ability of readers to process hundreds of chip reads virtually simultaneously. There are three main types of RFID tags: Passive, Active and Semi-Passive.



Radio Frequency Identification (RFID), Figure 2 RFID chip surrounded by its long antenna coil. (Source: Defense Logistics Agency (DLA))

Passive Tags Passive RFID tags do not have their own power supply. They harvest necessary power from the electromagnetic fields emanating from the incoming radio wave transmissions they receive from RFID readers. As transponders they 'awake' on demand by the reader's signal and induce a small current in the on board antenna to power-up the internal (CMOS) microchip and send back data to the reader. The term 'backskatter' is used to describe the process by which the tags reflect the signals they receive from readers as their message-laden reply. The reflected signals are modulated variations of the signals that are received and therefore produce meaningful data- the difference between the original and the (modulated) variant. Today's chips hold up to about two kilobytes of data, enabling the storage of rich information about the tagged entity. Because passive tags do not require on-board batteries they have a significantly longer life expectancy



Radio Frequency Identification (RFID), Figure 3 Passive RFID label. (Source: DLA)

than active tags, can be made much smaller and are therefore cheaper to produce. Unit costs vary with capabilities and purchase volume but the lowest cost passive tags currently sell for about twenty U.S. cents per unit in high volume purchases. New non-silicon (polymer) roll-printable tags are currently being prototyped and these should further reduce the manufacturing cost. Typical sizes range from postage stamps to postcards though the smallest passive tags are thinner than a standard piece of paper and approximately three millimeters across and therefore can be inserted under the skin of animals and people or fit nicely on a sticker or paper label (Fig. 3). The lack of internal power creates a limitation on broadcast distance and thus passive tags are generally used for short distance transmissions (under 10 feet). Passive tags are typically write-once-read-many (WORM) devices, meaning that their memory cannot be overwritten.

Active Tags Unlike their passive counterparts that employ backscatter to send message replies to readers, active tags have an internal transmitter that allows them to send waves under their own power and conduct what amounts to 'sessions' with readers. Most have on-board batteries, though some use solar and other power supply options to harvest energy to power the transmitter. The resulting larger form factor and higher cost is offset by the distinct benefit of being able to transmit stronger signals, useful in situations where required read distances are much further as well as in environments that are difficult for passive tags to function well (such as those high in metals or water). Active tags often have readable ranges of hundreds of meters and vary in price from a few dollars up to twenty dollars or more per unit. Life expectancy of active RFID chips tends to be based on battery life and is typically up to 10 years. In addition to environments that challenge passive tags, active RFID tags are often used to track valuable assets. They usually have more data storage capability than passive tags and may also provide read-write capabilities so that internally stored data can be updated and changed. Active tags are frequently combined with environmental sensors to produce sophisticated monitoring devices that can, for example, track a product in its shipping route while monitoring the temperature and other conditions in which it is being transported.

Semi-Passive Tags Semi-passive tags are a hybrid of active and passive tags. They have on-board battery power like active tags but they use backscattering to reflect radio waves back toward the readers that produce them, rather than transmitting their own message broadcasts. The battery is used to power the microchip and (optionally) environmental sensors. While their versatility exceeds that of passive tags they are cheaper than active tags and their low battery consumption means they live longer.

Readers

In an RFID system the readers are the components that directly communicate with and fetch data from tags. Readers may be stationary or mobile (Fig. 4) and have one or more antennas which broadcast signals to nearby tags, querying information stored in the embedded chips as well as (occasionally) from on-board sensors. Agile readers are highly functional and can read tags at different frequencies and use multiple communication protocols in communicating with various tags. Some readers (especially those

Radio Frequency Identification (RFID), Figure 4 Mobile RFID reader with tags in foreground. (Source: PM J-AIT)

in the ultra-high frequency range) can have a 'null spot', a blind-spot of sorts for an RFID reader. There are two primary types of RFID readers: **read-only** and **read-write**. As implied by the name, the former class is limited to reading and extracting data from tags. Some in this category are referred to as 'dumb' due to limited computing capabilities. Dumb readers convert the radio waves they receive from RFID tags into binary values and simply forward the converted values to a server for processing of actual business logic and filters. Unlike read-only devices, the latter class of readers can read from RFID tags as well as write to them. The roles of read-write readers include initializing chips on new WORM tags and writing to updateable chips during RFID transactions.

Air Interface Protocol In any field the work of establishing and refining protocols and standards is long term and ongoing. An area of obvious focus in RFID is the communication between readers and tags. Whether articulating the differences between *tag talks first* (new tags crossing a reader's threshold 'announce' themselves to the reader) and *reader talks first* (readers initiate contact with tags by sending out inquiries to all listening tags) protocols, following mutual encoding rules, establishing system commands for reading and writing data or knowing the rules for how tags should modulate responses back to readers, the standards and protocols are critical to the success of RFID. Among the many standards governing the RFID industry the air interface protocol provides specific directives concerning the communication between readers and tags. As of July 2006 the new 'Gen 2' air-interface protocol was incorporated into the ISO 18000–6 standard [19].

Singulation and Anti-Collision One of the specific technical challenges of RFID readers is identifying unique tags amongst many tags within a given read field and being able to gracefully handle the resulting collisions from simultaneous radio responses by multiple tags (singulation). This is a common problem in environments such as warehouses and grocery stores, where RFID readers routinely have hundreds of tags within close proximity. One commonly used singulation protocol is known as 'tree walking.' The RFID reader logically traverses a binary tree in a recursive, depth-first algorithm. Each node has only two children: 0 for the left subtree and 1 for the right subtree (the root node is considered empty). The reader broadcasts each successive bit sequence (i. e. tree node label) to all of the tags within its range, instructing only those tags with serial numbers matching the broadcast sequence to respond (by providing their next serial number bit), while the rest of the tags remain silent. When collisions result between responses from both the left and right subtrees of the broadcast node, the reader detects the collision and interprets it as a requirement to descend into both subtrees (recursively and sequentially). The algorithm ensures that the reader descends into every required subtree (i. e. those containing currently present leaf nodes). As the reader recurses through the tree, it captures the leaf nodes bit strings, resulting in a list of all (binary) serial numbers of the individual tags in its range [18]. Though it is widely used, the tree-walking protocol exposes RFID systems to substantial threats of privacy and security. The identify of tags within range of the reader can be deduced by eavesdropping on only one side of the electronic conversation (the reader side) and the longer broadcast range of readers enables such eavesdropping to be conducted at distances well beyond the range of the tags themselves.

Label Printers

The RFID industry distinguishes between tags and labels: transponders mounted to a substrate for the purpose of attaching to a carton or other surface are considered RFID tags. Transponders with the smallest (thinnest) form factor that are bound with adhesive between pieces of paper in order to be used in labels are called RFID labels. RFID label printers are highly specialized printing machines that output 'smart' labels which combine bar code printing technology with RFID. The printers simultaneously print bar codes on the front of the labels while writing (related) data to the on-board chips of RFID transponders embedded between the sheets of label paper (Fig. 5).

Radio Frequency Identification (RFID), Figure 5 Smart label printer combining bar codes and RFID tags. (Source: DLA)

Key Applications

RFID is broadly utilized across a wide array of identification systems, addressing such problems as inventory control, tracking and supply chain management, passport,

currency and border control, transportation access and automated fare/toll collection, hands-free ski lift access, bovine control, wildlife tracking and pet identification, building access control, performance monitoring and sensor reading, prisoner tracking, hospital bed management and crime (theft, tampering, counterfeiting) deterrence. The technology has become ubiquitous in modern society and is embedded in the products sold by mass retailers, the currency and credit cards used for payment, vending machines, library systems, warehouse pallets, airline baggage handling machines, automobile tires, keys, sports match timing devices, animals and people.

Radio Frequency Identification (RFID), Figure 6 Truck transponder sends data to roadside RFID reader via overhead antennae. (Source: FHWA)

Inventory Control

In 2003 mega-retailer Wal-Mart single-handedly initiated a major industrial trend when it announced it would be requiring top suppliers to implement RFID technology for inventory management. Others were quick to follow the world's largest company down the RFID path, including the Department of Defense, Target Stores, Best Buy, Kroger, Circuit City and others. Prior to this, one of the biggest obstacles to adoption was the relatively high cost of producing tags but by throwing its (unparalleled) business volume behind the technology Wal-Mart played a significant role in driving down cost and ensuring RFID's long term viability. In inventory management systems RFID tags are affixed to pallets, boxes and even individual products to facilitate inventory tracking with little required human intervention. For example entire cartons of products can be added to the inventory pool without requiring they be scanned individually or even removed from the box. In the long term, RFID may also be used to track the movement of products through stores. Though Wal-Mart has lagged behind its original goal of having its 100 top suppliers converted by January 2005 it continues to push ahead and recently announced it would double the number of its stores that have implemented RFID and that the number would reach 1000 locations by the beginning of 2007 [35].

Supply Chain Management

Wal-Mart's introduction of RFID into the retail space was echoed by a similar move at the U.S. Department of Defense (DOD) to transition its supply chain management structure to an RFID-based system. Over the past several years the DOD has implemented RFID on a broad, international scale across its operations. One of the interesting RFID issues in supply chain management is tag ownership. Unlike in-house applications where a single entity invests in and maintains a set of tags for RFID, a supply chain by definition contains many actors and goods pass through many boundaries of ownership and control as they traverse the chain from source to destination (Fig. 6). In such an environment, tags simply become part of the shipping material and are rarely recycled or reused by the originating party. These are precisely some of the requirements driving tagging technology toward extremely low cost tags that can become as ubiquitous and disposable as bar codes are today. Tag disposal and recycling are related concerns that take on new meaning in such cooperative systems. Finally, just as the lines of tag ownership blur in supply chain applications, so too does the ownership of technology standards. Privately controlled proprietary standards do not meet the requirements of systems punctuated by repeated hand-offs from one party to another. Applications such as supply chain management will compel the state of RFID technology to continue to move toward open industry standards.

Passport, Currency and Border Control

Another key RFID application is in the area of protecting national assets such as passports, currency and physical borders. RFID passport control systems already exist in many countries. The U.S. has had an off-again on-again relationship with passport-embedded RFID but as of October 2006 is now issuing all new passports with ISO 14443 RFID tags. Security is an obvious and critical requirement for such systems yet this is an area where RFID has struggled. In the Netherlands in early 2006 the Dutch security company Riscure demonstrated a successful compromise of a Dutch RFID-passport based on the same ISO standard [36]. Even human-embedded RFID tags have been unable to escape security controversy. In July of 2006 hackers at a conference in New York City demonstrated the cloning of an 'uncloneable' implanted RFID chip and successfully fooled the RFID reader concerning the identity of the bearer [22]. Another area of broad concern has been the tracking of national currencies for the purposes of reducing fraud and criminal enterprise. While the com-

plete materialization of a plan may be years away, there has been a great deal published in both academia and industry on the specter of embedded RFID tags in European currency [21]. Finally, while contact-less toll collection systems have been in place for many years, they are now being applied to certain border crossings between the US and Canada. Completely automated border security may not be realistic, but in such cases RFID is facilitating speedy identification of those wishing to pass national boundaries.

Future Directions

As seen in the previous section, radio frequency identification is a technology with seemingly limitless potential for future growth. When combined with GIS, that landscape of future possibilities appears to grow without bound. Systems are being pioneered and developed today that bring the two technologies together in interesting ways such as in wildlife management, military field training and exercises, monitoring of children through embedded RFID/GIS devices in school bags, automated manhole monitoring and intelligence, remote forestry management, election support and targeted voting campaigns, 'pervasive retail' and 'smart' environments that are aware of and respond to entities passing through them. However, as RFID and GIS converge with other technologies such as bar coding, digital cameras and biometrics, concerns about security and privacy will continue to grow. Issues such as data privacy and identity theft, RFID tag blocking, data leaking and RFID viruses will grow in prominence and public awareness. RFID's impact on the environment in such areas as device disposal and increased radio wave transmissions will likely become a major area of emphasis, as will increased efforts toward further standardization of worldwide RFID systems. Finally, one would expect that as systems begin to mature and require less break-fix attention the systematic mining of considerable amounts of newly captured data will likely become a major area of increased research and emphasis.

Cross References

- ► Emergency Evacuation Plan Maintenance
- ► Emergency Evacuations, Transportation Networks
- ► Homeland Security and Spatial Data Mining
- ► Information Services, Geography
- ► Privacy and Security Challenges in GIS

Recommended Reading

1. Landt, J.: Shrouds of Time: The History of RFID. http://www.aimglobal.org/technologies/rfid/resources/shrouds_of_time.pdf, Association for Automatic Identification and Mobility: Retreived 10/10/2006 (2001)
2. Wikipedia: Radio Frequency Identification. http://en.wikipedia.org/w/index.php?title=Radio_Frequency_Identification&oldid=85251329, Wikipedia, The Free Encyclopedia: Retreived 10/10/2006 (2006)
3. Vogt, H.: Pervasive Computing: First International Conference Proceedings, In: Efficient Object Identification with Passive RFID Tags, Lecture Notes in Computer Science, vol. 2414, pp. 98. Springer (2002)
4. The British Broadcasting Corporation: Stopping the book thieves. http://news.bbc.co.uk/2/hi/uk_news/1958424.stm, The BBC News: Retreived 10/10/2006 (2002)
5. Shim, R.: Wal-Mart to throw its weight behind RFID, http://news.com.com/2100-1022-1013767.html?tag=nl, CNET News: Retreived 10/10/2006 (2003)
6. Bauer, M., Fabian, B., Fischmann, M., Gürses, S.: Emerging Markets for RFID Traces, Arxiv preprint cs.CY/0606018 (2006)
7. Bluestein, G.:, Companies track gridlock via cell phones. http://news.yahoo.com/s/ap/20061105/ap_on_hi_te/tracking_traffic;_ylt=AoQQdZOHq.vxPTOcxWTRCaDMWM0F;_ylu=X3oDMTA3ODdxdHBhBHNlYwM5NjQ-, Associated Press: Retreived 11/05/2006 (2006)
8. Saxena, M.: Does GIS and RFID Amalgamation Work? http://www.geoplace.com/uploads/FeatureArticle/0505tt.asp, Geoplace: Retreived 10/10/2006 (2005)
9. Stockman, H.: Communication by means of reflected power, Proc. Inst. Radio Engin. **36**, 1196–1204 (1948)
10. The United States Navy: Identification Friend or Foe. http://www.nrl.navy.mil/content.php?P=FRIENDFOE, United States Naval Research Library: Retreived 11/05/2006 (2006)
11. Bowden, L.: The story of IFF (identification friend or foe), IEEE Proceedings, Part A: Physical Science, Measurement and Instrumentation, Management and Education **132**(6), 435–437 (1985)
12. Vernon Jr, F.: Application of the microwave homodyne, Antennas and Propogation, Transactions of the IRE Professional Group on **4**(1), 110–116 (1952)
13. Walton, C.A.: US Patent 3,752,960, 27 December 1971
14. The RFID Journal: The History of RFID Technology. http://www.rfidjournal.com/article/articleview/1338/1/129/, The RFID Journal: Retreived 10/10/2006 (2006)
15. Koelle, A.R., Depp, S.W., Freyman, R.W.: Short-range radio-telemetry for electronic identification, using modulated RF backscatter, Proc. IEEE **63**(8), 1260–1261 (1975)
16. Sanjay E. Weis, S., Weis, S.A., Engels, D.W.: RFID Systems and Security and Privacy Implications, In: Workshop on Cryptographic Hardware and Embedded Systems, Lecture Notes in Computer Science, vol. 2523, pp. 454–470 (2002) http://citeseer.ist.psu.edu/sarma02rfid.html
17. Molnar, D., Wagner, D.: Privacy and security in library RFID: issues, practices, and architectures, In: CCS '04: Proceedings of the 11th ACM conference on Computer and communications security, pp.210–219. ACM Press, New York (2004)
18. Juels, A., Rivest, R.L., Szydlo, M.: The blocker tag: selective blocking of RFID tags for consumer privacy, In: CCS '03: Proceedings of the 10th ACM conference on Computer and communications security, pp. 103–111. ACM Press, New York (2003)
19. O'Connor, M.C.: Gen 2 EPC Protocol Approved as ISO 18000–6C. http://www.rfidjournal.com/article/articleview/2481/1/1/, The RFID Journal: Retreived 11/05/2006 (2006)
19. Avoine, G., Oechslin, P.: Financial Cryptography and Data Security, In: RFID traceability: A multilayer problem, Lecture Notes in Computer Science, vol. 3570, pp. 125–140. Springer (2005)

21. Juels, A., Pappu, R.: Financial Cryptography, In: Squealing Euros: Privacy Protection in RFID-Enabled Banknotes, Lecture Notes in Computer Science, vol. 2742, pp. 103–121. Springer (2003)
22. Fulton, N.: High-tech cloning. http://blogs.reuters.com/2006/07/22/high-tech-cloning/, Reuters: Retreived 10/10/2006 (2006)
23. Weis, S.A., Sarma, S.E., Rivest, R.L., Engels, D.W.: Security in Pervasive Computing, In: Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems, Lecture Notes in Computer Science, vol. 2802, pp. 201–212. Springer (2004)
24. Indulska, J., McFadden, T., Kind, M., Henricksen, K.: Distributed Applications and Interoperable Systems, In: Scalable Location Management for Context-Aware Systems, Lecture Notes in Computer Science, vol. 2893, pp. 224–235. Springer (2003)
25. Roussos, G., Kourouthanasis, P., Spinellis, D., Gryazin, E., Pryzbliski, M., Kalpogiannis, G., Giaglis, G.: Systems architecture for pervasive retail, In: SAC '03: Proceedings of the 2003 ACM symposium on Applied computing (ACM Press, New York 2003)
26. Newsweek: An Internet of Things. http://www.msnbc.msn.com/id/3068871/, Newsweek: Retreived 10/10/2006 (2006)
27. Davies, N., Gellersen, H.W.: Beyond prototypes: challenges in deploying ubiquitous systems, Pervasive Computing, IEEE **1**(1), 26–35 (2002)
28. Tenmoku, R., Kanbara, M., Yokoya, N.: A wearable augmented reality system using positioning infrastructures and a pedometer, Wearable Computers, 2003. Proceedings. Seventh IEEE International Symposium on 110–117 (2003)
29. Want, R., Fishkin, K.P., Gujar, A., Harrison, B.L.: Bridging physical and virtual worlds with electronic tags, In: CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems. ACM Press, New York (1999)
30. Coroama, V., Bohn, J., Mattern, F.: Living in a Smart Environment – Implications for the Coming Ubiquitous Information Society, In: Proceedings of the International Conference on Systems, Man and Cybernetics 2004 (IEEE SMC 2004) 5633–5638, http://citeseer.ist.psu.edu/coroama04living.html
31. Juels, A., Weis, S.A.: Defining Strong Privacy for RFID. http://eprint.iacr.org/2006/137.pdf, Cryptology ePrint Archive: Retreived 10/10/2006 (2006)
32. Roberti, M.: DOD Releases Final RFID Policy. http://www.rfidjournal.com/article/articleview/1080/1/1/, The RFID Journal: Retreived 10/11/2006 (2004)
33. The RFID Journal: The History of RFID Technology. http://www.rfidjournal.com/article/articleview/1337/1/129/, The RFID Journal: Retreived 10/12/2006 (2006)
34. The RFID Journal: RFID System Components and Costs. http://www.rfidjournal.com/article/articleview/1336/1/129/, The RFID Journal: Retreived 10/12/2006 (2006)
35. The RFID Gazette: Wal-Mart Doubling RFID-Enabled Stores. http://www.rfidgazette.org/walmart/, The RFID Journal: Retreived 11/05/2006 (2006)
36. Riscure: Privacy issues with new digital passport. http://www.riscure.com/2_news/passport.html, Riscure BV: Retreived 11/08/2006 (2006)

Radio Tagging

► Radio Frequency Identification (RFID)

Radiolocation

► Indoor Positioning with WirelessLocal Area Networks (WLAN)

Range Combining

► Indoor Localization

Range Query

► Nearest Neighbors Problem
► R*-tree

Range Query Algorithm

► Indexing of Moving Objects, B^x-Tree

Ranking, Multi-Dimensional

► Skyline Queries

Ranking Results

► Skyline Queries

Raster

► Spatial Data Transfer Standard (SDTS)

Raster Data

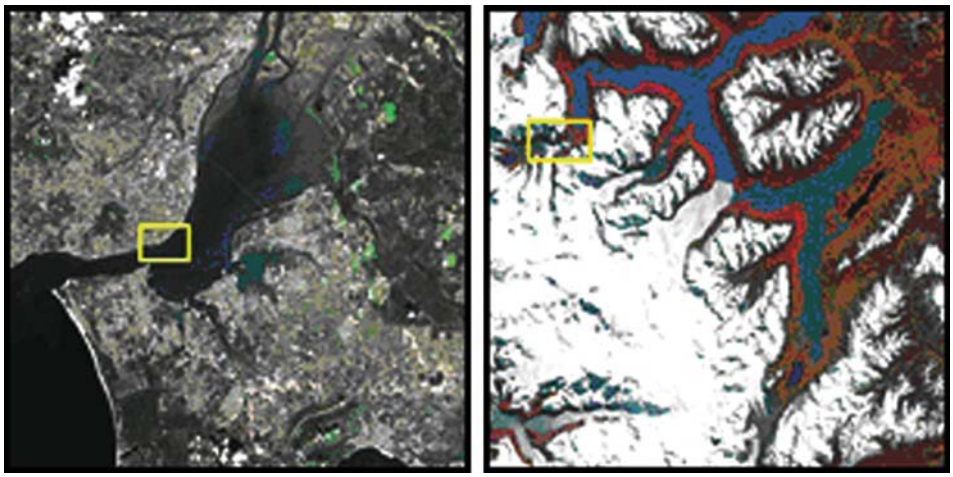
HYEEUN LIM
Department of Computer Science & Engineering, University of Minnesota, Minneapolis, MN, USA

Synonyms

Raster image; Digital image; Bitmap

Definition

Raster displays and databases build all geographic features from grid cells in a matrix. A raster display builds an image from pixels, or elements of coarse or fine resolution. A raster database maintains a similar "picture" of reality in which each cell records some sort of information averaged over the cell's area. The size of the cell may again be coarse or fine, ranging from centimeters to kilometers.



Raster Data, Figure 1
Imagery samples (5 m) from SPOT-5 [4]

Basically, a raster file is a giant table, where each pixel is assigned a specific value from 0 to 255. The meaning behind these values is specified by the user. They can represent elevations, temperatures, hydrography, etc. Satellite imagery uses raster data to record different wavelengths of light. Many satellites, like Landsat and SPOT (Fig. 1), transmit raster images of the earth's surface. Raster data is generally divided into two categories: thematic data and image data. The values in thematic raster data represent some measured quantity or classification of particular phenomena such as elevation, pollution concentration or population. For example, in a landcover map the value 5 may represent forest, and the value 7 may represent water. The values of cells in an image represent reflected or emitted light or energy such as that of a satellite image or a scanned photograph.

Historical Background

The earliest geographic information systems (GIS) architectures, implemented by Roger Tomlinson in the Canadian Land Inventory in the mid 1960s, emulated traditional map drafting. Entities were represented by points and lines that could be drawn with an automated drafting machine (aka pen plotter.) An outline history of GIS can be found at National Center for Geographic Information and Analysis History of GIS.

At The Harvard Lab for Computer Graphics and Spatial Analysis [at the Harvard Graduate School of Design (GSD)] in the late 1960s and early 1970s, Carl Steinitz and many others were experimenting with ways to use digital geographic data to emulate the cartographic overlay techniques portrayed by Ian McHarg in his book *Design with Nature*. McHarg and Steinitz collaborated in the first digitally augmented landscape planning studio at the GSD in 1967.

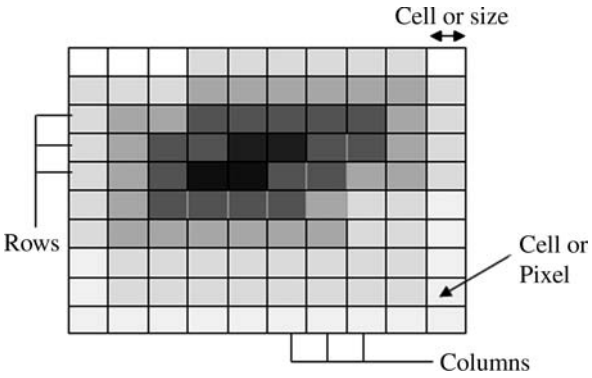
The students and researchers at Harvard were using a computer language called FORTRAN, IBM's highly customizable information tool to represent landscapes and things that may happen on them. In FORTRAN, one of the fundamental forms of representation is a matrix: an array of values. Naturally, someone began to experiment with this structure to represent the character of locations in 2-D space, and Raster GIS was born.

In general, raster data are more suited to environmental applications while vector data are more suited to human activity. Raster systems model complex spatial patterns with limited attributes, such as land-use patterns very well, while the vector data model is better for more clearly defined space with complex attributes, such as census data. There are exceptions to this: Martin (1996) uses derived raster surfaces to model 1981 and 1991 census data and compare change between the two, claiming that the raster model provides a more realistic model of the underlying population distribution. A good example of a raster system in a historical context is provided by Bartley and Campbell (1997). They examined the Inquisitions Post-Mortem of the fourteenth century and used these to create a raster GIS of medieval land-use that, they claim, is potentially the most detailed survey possible until the nineteenth century tithe surveys. A raster system was used because it provides a complete coverage of the land area, it provides a more realistic representation of land use than polygons, and because it handles source inaccuracies better than polygons.

Scientific Fundamentals

Raster Data Structure

Raster data types consist of rows and columns of cells, in each of which is stored a single value. Cells are arranged



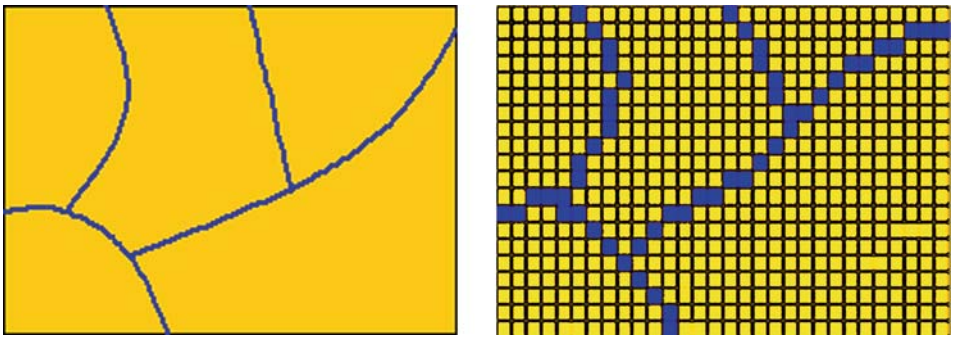
Raster Data, Figure 2 Raster data structure

in rows and columns where rows represent the *x*-axis of a Cartesian plane and the columns the *y*-axis. Each cell has a unique row and column address. Figure 2 shows the raster data structure. Each cell must be rectangular in shape, but not necessarily square. Each cell within this matrix contains location coordinates as well as an attribute value. Values can represent magnitude, distance, or relationship of the cell on a continuous surface. Values can also represent categorical data such as soil type or land-use class. Both integer and floating-point values are supported in spatial analyst. Integer values are best used to represent categorical data and floating-point values to represent continuous surfaces such as elevation, slope or flow accumulation. Rastering is a method to store, process and display spatial data. Most raster data are images, but in addition to color, the value recorded for each cell may be a discrete value, such as land use, a continuous value, such as rainfall, or a null value if no data is available.

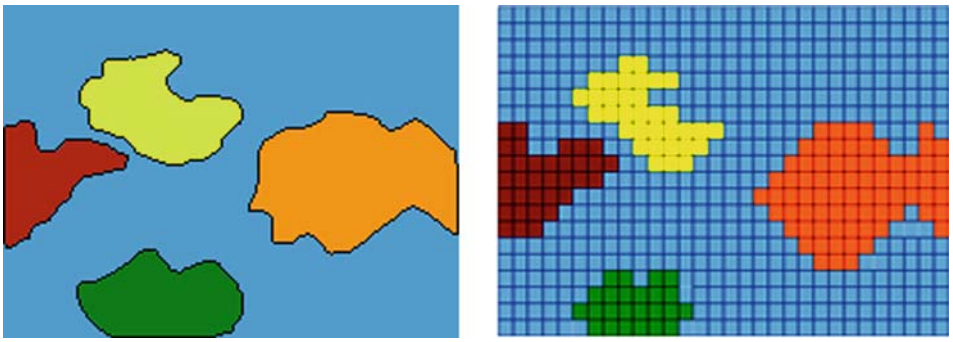
The spatial location of each cell is implicitly contained within the ordering of the matrix, unlike a vector structure which stores topology explicitly. Areas containing the same attribute value are recognized as such; however, raster structures cannot identify the boundaries of such areas as polygons. While a raster cell stores a single value, it can be extended by using raster bands to represent red, green and blue(RGB) colors, colormaps (a mapping between a thematic code and RGB value), or an extended attribute table with one row for each unique cell value. Thus, these two or more cells with the same value belong to the same zone. A zone can consist of cells that are connected, disconnected, or both. Zones are not analagous to polygons, which are only contiguous closed areas.

Representing Features in a Raster Dataset

Linear Data Linear data is all of those features that, at a certain resolution, appear only as a polyline such as a road, a stream, or a power line. A line by definition does not have area. A polyline can be represented only by a series of connected cells (Fig. 3). As with a point, the

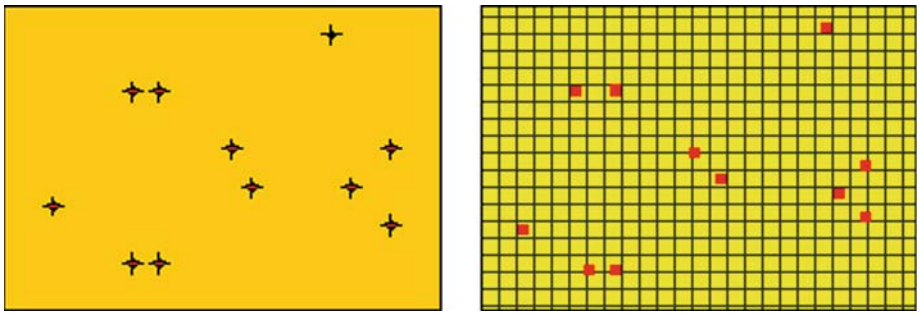


Raster Data, Figure 3
A set if line features represented in a grid



Raster Data, Figure 4
A set of polygon features represented in a grid

R



Raster Data, Figure 5 A set of point features represented in a grid

accuracy of the representation will vary according to the scale of the data and the resolution of the raster dataset.

Polygon Data Polygonal or aerial data is best represented by a series of connected cells that best portrays its shape (Fig. 4). Trying to represent the smooth boundaries of a polygon with a series of square cells does present some problems, the most infamous of which is called the "jaggies", an effect that resembles stair steps.

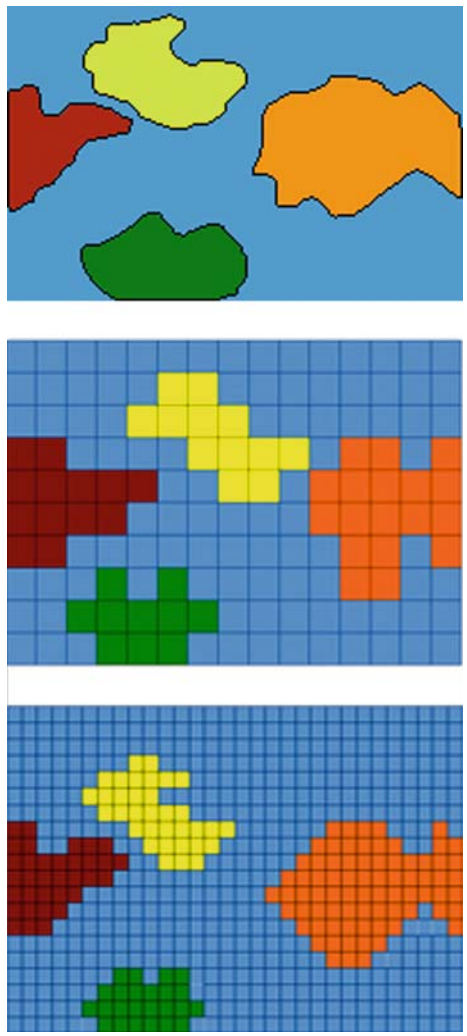
Point Data A point feature is any object at a given resolution that can be identified as being without area. Although a well, a telephone pole, or the location of an endangered plant are all features that can be rendered as points at some resolutions, at other resolutions they do, in fact, have area. Point features are represented by the smallest unit of a raster, a cell (Fig. 5). It is important to remember that a cell has area as a property. The smaller the cell size, the smaller the area and thus the closer the representation of the point features. Points with area have an accuracy of plus or minus half the cell size. This is the trade-off that must be made when working with a cell-based system. Having all data types – points, polylines, and polygons – in the same format and being able to use them interchangeably in the same language are more important to many users than a loss of accuracy. The accuracy of the above representation is dependent on the scale of the data and the size of the cell.

The Resolution of a Raster Dataset

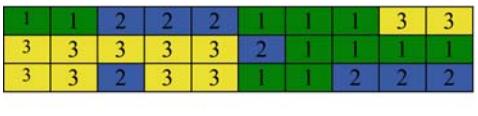
The size chosen for a raster cell of a study area depends on the data resolution required for the most detailed analysis (Fig. 6). The cell must be small enough to capture the required detail, but large enough so that computer storage and analysis can be performed efficiently. The more homogeneous an area is for critical variables such as topography and land use, the larger the cell size can be without affecting accuracy. Before specifying the cell size, the following factors should be considered:

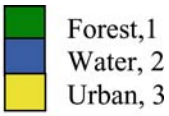
- The resolution of the input data
- The size of the resultant database and disk capacity
- The desired response time
- The application and analysis that are to be performed

A cell size which is finer than the input resolution will not produce more accurate data than the input data. It is generally accepted that the resultant raster dataset should be the



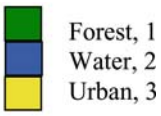
Raster Data, Figure 6 An example of different resolution raster data maps





⇒1122211133
3333321111
3323311222
=30

Raster Data, Figure 7 An example of exhaustive enumeration



⇒21 32 31 23
53 12 41
23 12 23 21 32
= 24

Raster Data, Figure 8 An example of run-length encoding

same or coarser than the input data. Spatial analysis allows for raster datasets of different resolutions to be stored and analyzed together in the same database.

Raster Data Compression Technique

Exhaustive Enumeration Every pixel is given a single value; there is no compression when many like values are encountered (Fig. 7).

Run-Length Encoding This is a raster image-compression technique. If a raster contains groups of cells with identical values, run-length encoding can compress storage (Fig. 8). Instead of storing each cell, each component stores a value and a count of cells with that value. If there is only one cell, the storage doubles, but for three or more cells, there is a reduction. The longer and more frequent the consecutive values, the greater the compression that will be achieved. This technique is particularly useful for encoding monochrome images or binary images.

Raster Data Versus Vector Data

Vector Data The vector data type uses geometries such as points, lines or polygons to represent objects. Examples include property boundaries for a housing subdivision represented as polygons and well locations represented as points. Vector features can be made to respect spatial integrity through the application of topology rules such as polygons not being allowed to overlap. Vector data can also be used to represent continuously varying phenomena. Contour lines and triangulated irregular networks (TINs) are used to represent elevation or other continuously changing values. TINs record values at point locations, which are connected by lines to form an irregular mesh of triangles. The face of the triangles represents the terrain surface.

Comparison Between Vector Data and Raster Data See Table 1 and Fig. 9

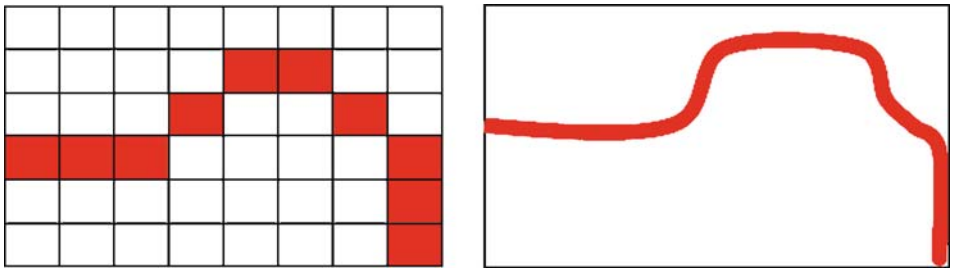
Key Applications

Business Mapping/Geomarketing

Business mapping improves the understanding and control of the market's realities and opportunities through the connections that it creates between data and the corresponding geographical location. Geomarketing, a derivative of business mapping, is the branch of marketing that helps to model and analyze a set of factors in order to find out correlations between a consumer's dwelling location and workplace, as well as his consumption patterns. Thematic raster data can be used for business mapping and geomarketing in order to optimize business actions by favoring geographical criteria through the use of statistical analysis and cartography (Fig. 10).

Health Geography

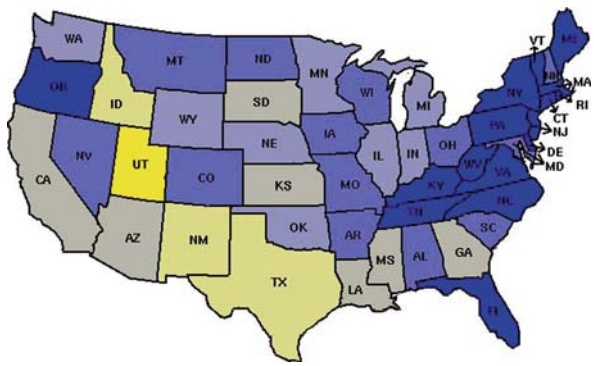
Raster data plays a major role in several areas of health geography. The cartographic representation of health data helps to describe and analyze a situation, and clarify it in complex cases. But the true purpose of health geography is to determine the reasons behind the phenomena. It is also important not to ignore the potential for new interactions with other fields of activity when geography is



Raster Data, Figure 9
A raster image compared with a vector image

Raster Data, Table 1 Comparison between vector data and raster data

	Raster data	Vector data
Character-istics	Simple "grid" structure of rows and columns. Based on cells or picture pixels. Linear feature is a contiguous set of cells. Resolution based on size of grid (cell): the smaller the cell, the higher the resolution. Features are considered homogenous within a pixel. Storage increases with the square of the resolution	Based on objects (points, lines, areas). Constructed using arcs, nodes and vertices. Resolution can be independent of detail. Every point has a unique location
Advantages	A simple data structure. Overlay operations are straight forward. High spatial variability is efficiently represented Only raster can store image data (e. g., photos)	Compact data structure for homogenous areas. Efficient encoding of topology. Better suited for map output
Disad-vantages	Data structure is not compact (though it can be modified). Topological relationships are harder to represent. Map output can appear "blocky"	More complex data structure. Overlay operations are more complex. High spatial variability is less efficiently stored. Cannot store (continuously varying) image data



Raster Data, Figure 10 An example of thematic maps

used dynamically in the health care system since it adds demographic, economic, social, cultural and environmental dimensions. The application of raster thematic data to health geography for analysis and decision making is an area with great potential in today's information and communication age.

Satellite Imagery

Satellite imagery is always in a raster format in which each pixel has its own signature. Reflectance at a certain wavelength is measured for each cell in an image. The cells may cover areas on the earth's surface several hundreds of meters square, the area covered being a function of a particular satellite's resolution. For example, Landsat, SPOT and digital raster graphics (DRG) are types of satellite data. Landsat, or thematic mapped data, was one of the first types of satellite data available to the public. It gives, at best, 10-m resolution although there is discussion that the new satellites to be launched soon will improve on that. Landsat, or thematic mapped data is very cost effective for large areas, particularly when we wish to acquire older scenes for an historic perspective. SPOT data is produced by a French company. It is a later form of the technology, and will give 3-m resolution. DRG data is generated from scanned quad maps. They can be very useful as backdrops to other data, for guiding digitizing, assisting in classification, or other operations where broad background coverage is needed.

Future Directions

Raster data is an abstraction of the real world where spatial data is expressed as a matrix of cells or pixels with spatial position implicit in the ordering of the pixels. Raster data can be applied to various areas such as business mapping, geomarketing, health geography or satellite imagery. Its usefulness for analysis and decision making will continue to grow in today's information and communication age.

Cross References

► Information Services, Geography
► Vector Data

Recommended Reading

1. Wikipedia http://en.wikipedia.org/wiki/Geographic_information_system Accessed 23 March 2007
2. Introduction to GIS–Raster-Based GIS. http://www.sli.unimelb.edu.au/gisweb/GISModule/GIST_Raster.htm Accessed 21 February 2007
3. Understanding Raster Data And Cell-Based Modeling. http://www.cfr.washington.edu/research.urbaneco/GIS_Tutorial Accessed 21 February 2007
4. Geo VAR-Geospatial Data and Information System http://www.geovar.com/data/satellite/spot/spot.htm?gclid=CNHD6OS8ooQCFQ5vOAodVR8rgg Accessed 23 March 2007
5. Shekhar, S., Chawla, S.: Spatial Databases: a Tour. Prentice Hall (2003)
6. Raster GIS http://www.gsd.harvard.edu/pbcote/courses/gsd6322/raster/index.htm Cambridge, MA, USA
7. ADHS Guides to Good Practice http://ahds.ac.uk/history/creating/guides/gis/ Accessed 21 March 2007

Raster Data Compression

► Data Compression for Network GIS

Raster Image

► Raster Data

Raster Models

► Data Models in Commercial GIS Systems

Rational Choice

► Crime Mapping and Analysis

Ra*-Tree

► Multi-Resolution Aggregate Tree

RCC

► Mereotopology

Realignment

► Conflation of Features

Real-Time Generalization

► Generalization, On-the-Fly

Real-Time Location Services

► Indoor Positioning

Reasoning

► Geospatial Semantic Web: Personalisation

Reasoning, Spatio-temporal

► Temporal GIS and Applications

Record-to-Record Travel Method

► Routing Vehicles, Algorithms

Rectangle, Hyper-

► Indexing, X-Tree

Rectangle, Minimum Bounding

► Indexing, X-Tree

Reference, Coordinate Based

► Indoor Localization

Reference, Symbolic

► Indoor Localization

Reference System, Spatial

► Geography Markup Language (GML)
► Oracle Spatial, Geometries

Reference System, Temporal

► Geography Markup Language (GML)

Reference-Feature Centric

► Co-location Patterns, Algorithms

Region, Conjeccture

► Vague Spatial Data Types

Region Connection Calculus

► Mereotopology

Regional Sience

► Time Geography

Regionalization, Spatial Ontologies

► Geodemographic Segmentation

Registration

JÉRÔME THÉAU
GIS Training and Research Center,
Idaho State University, Idaho, ID, USA

Synonyms

Coregistration; Co-registration; Georegistration

Definition

Registration is the process of geometrically matching different spatial datasets of the same area so that identical features are perfectly superimposed. In GIS, this match is usually performed within common geographic coordinate system. Spatial datasets may be from different sensors (e. g. satellite, aerial) and sources (e. g. maps, images), and acquired at different times (e. g. multidate satellite imagery).

Main Text

Registration is usually performed using one dataset as a reference (or master) and the other datasets as targets (or slaves) to correct. Registration can also be independently performed on different datasets (e. g. independent registration of multidate images to a common projection). In this case, registration does not involve any reference-target processing.

Basically, registration algorithms can be linear or non-linear. Linear transformations involve translation, rotation, or scaling of the target dataset using, for example, first order transformations. Non-linear transformations are performed to correct for local and non-homogeneous deformations using, for example, polynomial transformations.

Registration algorithms are usually based on feature selection to perform the transformation, such as points, line intersections, or boundaries. However, some algorithms also use the structure of the image to perform the registration using, for example, Fourier analysis.

Registration is a critical step because many GIS applications such as change detection studies require two or more datasets of the same area. Accurate registration is therefore necessary to obtain accurate results.

Cross References

► Change Detection
► Co-location Pattern Discovery
► Correlation Queries in Spatial Time Series Data

Registry Information Model

► Catalogue Information Model

Regression

► Spatial and Geographically Weighted Regression

Regression, Geographically Weighted

► Geodemographic Segmentation

Reinsert, Forced

► R*-tree

Relative Location

► Localization, Cooperative

Relative Positional Accuracy

▶ Positional Accuracy Improvement (PAI)

Relevance, Spatial

▶ Retrieval Algorithms, Spatial

Relevance, Textual

▶ Retrieval Algorithms, Spatial

Reliable Real-Time Data Collection

▶ Data Collection, Reliable Real-Time

Remote Sensing

▶ Photogrammetric Sensors
▶ Standards, Critical Evaluation of Remote Sensing

Remote Sensing, Aerial

▶ Evolution of Earth Observation

Remote Sensing, Satellite-Based

▶ Evolution of Earth Observation

Remote Sensing Specifications

▶ Standards, Critical Evaluation of Remote Sensing

Remote Sensing Standards

▶ Standards, Critical Evaluation of Remote Sensing

Representational State Transfer Services

▶ Web Services

Representing Regions with Broad Boundaries

▶ Representing Regions with Indeterminate Boundaries

Representing Regions with Indeterminate Boundaries

ANTHONY G. COHN
School of Computing, University of Leeds, Leeds, UK

Synonyms

Egg-yolk Calculus; Representing regions with broad boundaries

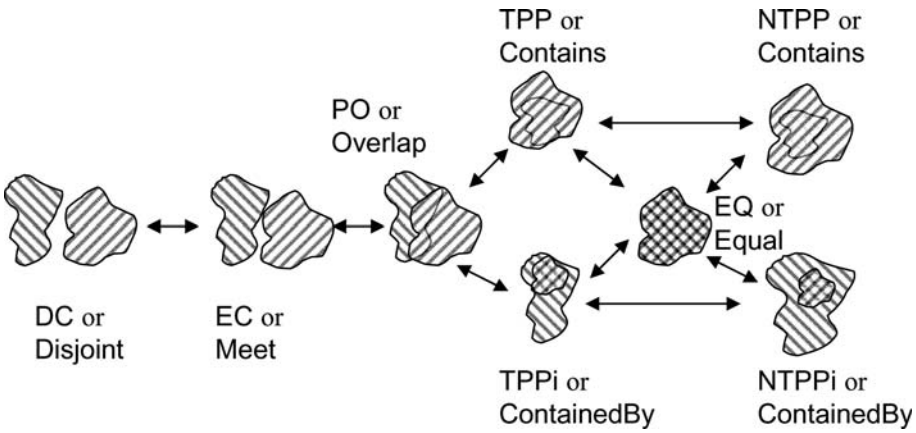
Definition

The problem of vagueness permeates almost every domain of knowledge representation. In the spatial domain, this is certainly true, for example it is often hard to determine a region's boundaries (e. g. "southern England"). Vagueness of spatial concepts can be distinguished from that associated with spatially situated objects and the regions they occupy. An adequate treatment of vagueness in spatial information needs to account for vague regions as well as vague relationships. Although there has been some philosophical debate concerning whether vague objects can exist,it is assumed here that they do, and some techniques for handling them are presented, specifically for considering the mereotopological relationships that may hold between such objects.

Historical Background

A number of approaches to representing and reasoning about regions with crisp boundaries had been developed by the mid 1990s [19,25] but the problem of treating regions with vague or indeterminate boundaries had not been specifically addressed. As a result of a workshop held in 1994 to investigate the problem of representing regions with indeterminate boundaries [7], two parallel but related calculi were proposed, each one based on one of the two main approaches to representing mereotopological relationships between two regions. The results [11,15] appear in in [7], and each approach has been further developed [12,16]. The approaches are also related to the notion of rough sets [6].

At least some of the same sorts of things about vague regions as about 'crisp' ones, with precise boundaries: that one contains another (southern England contains London, even if both are thought of as vague regions), that two overlap (the Sahara desert and West Africa), or that two are disjoint (the Sahara and Gobi deserts). In these cases, the two vague regions represent the space occupied by distinct entities, and we are interested in defining a vague area corresponding to the space occupied by either, by both, or by one but not the other. There might also be



Representing Regions with Indeterminate Boundaries, Figure 1 A 2D depiction of RCC-8 relations or the eight topological relations of the 4 and 9-intersection calculi. The arrows show the conceptual neighborhood structure

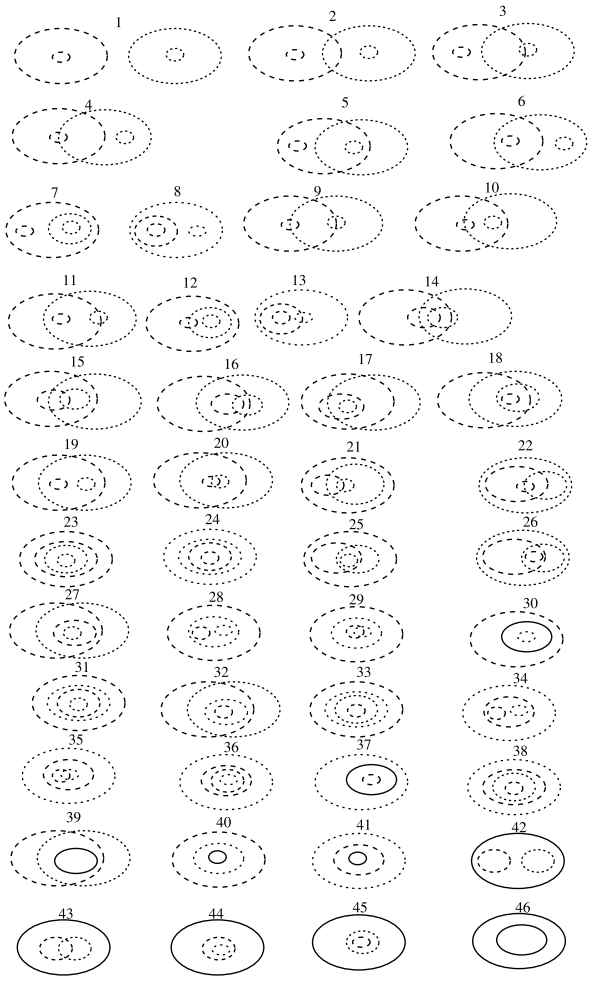
a requirement to say that one vague region is a 'crisper' version of another. For example, there might be an initial (vague) idea of the extent of a mineral deposit, and then information is received reducing the imprecision in the estimate of the deposit's extent. Here, the vagueness of the vague region is a matter of our ignorance: the entity concerned actually occupies a fairly well-defined region – though perhaps any entity's limits will be imprecise to some degree. In other cases, vagueness appears intrinsic: consider an informal geographical term like 'southern England'. The uncertainty about whether particular places (north of London but south of Birmingham) are included cannot be resolved definitively: it is a matter of interpretational context. A contrasting example is the region occupied by a cloud of gas from an industrial accident. Here are two sources of intrinsic vagueness: the concentration of the gas is likely to fall off gradually moving out of the cloud; and its extent will also vary over time, so any temporal vagueness (e. g., if asked about the cloud's extent 'around noon') will result in increased spatial vagueness.

In these cases of intrinsic vagueness, there is a degree of arbitrariness about any particular choice of an exact boundary, and often, none is required. But *if* it is decided to define a more precise version (either completely precise, or less vague but still imprecise), our choice of version is by no means *wholly* arbitrary: it is possible distinguish more and less 'reasonable' choices of more precise description. Distinguishing ignorance-based from intrinsic vagueness is important [2], but many of the same problems of representation and reasoning arise for both.

Scientific Fundamentals

There are two calculi in the literature which present sets of jointly exhaustive and pairwise disjoint mereotopological relations between pairs of regions. One is the so called Region Connection Calculus, which has two main variants, RCC-8 and RCC-5, with eight and respectively five *jointly exhaustive and pairwise disjoint* (JEPD) sets of relations. A 2D interpretation of the RCC-8 relations is depicted in Fig. 1; each of these relations can be defined within first order predicate calculus from a single primitive reflexive and symmetric relation **C**(*x*,*y*), true when region *x* is connected to region *y*. Alternatively, an essentially equivalent set of eight relations can be defined using the *4-intersection* method in which a relation between two regions *x* and *y* is defined in terms of whether the intersection of their boundaries and interiors is empty or non empty. An extension, the *9-intersection* method which also exploits the exterior of each region is also sometimes utilized. The RCC-5 calculus collapses the **DC** and **EC** relations to give a relation **DR**, the **TPP** and **NTPP** relations to give single proper part relation **PP** and similarly for **TPPi** and **NTPPi** giving **PPi**. The 4-intersection relations can be similarly reduced to five purely mereological relations.

These five relations have been extended to cover the case where the regions concerned do not have crisp boundaries. The extension of the RCC calculus [15,16] is called the "egg-yolk" calculus, while the extension of 9-intersection [11,12] is termed a spatial data model for objects with broad boundaries (henceforth SDMOBB). The egg-yolk calculus in fact originates in earlier work on data integration [23]. The approach in each of these is essentially the same – to identify a core region which always belongs to the region in question (the *yolk* in the terminology of former), and an extended region which might or might not be part of it (together forming the *egg*). There is a constraint that the yolk is always a proper part (**PP**) of the whole egg; the question then is: how many relations are there between the two regions, expressed in terms of how the respective yolks and eggs intersect? The answer is slightly different in the two cases, resulting from the two underlying construction techniques. In the egg-yolk case the RCC-5 relations are used to relate the components of each egg-yolk pair,



Representing Regions with Indeterminate Boundaries, Figure 2 The 46 relations of the egg-yolk calculus derived from RCC-5

and every consistent combination of RCC-5 relations over the 4 pairs of regions (egg_1-egg_2, egg_1-$yolk_2$, $yolk_1$-egg_2, $yolk_1$-$yolk_2$) is possible; it turns out there are 46 possible relations which are displayed in Fig. 2.

In the SDMOBB the 9-intersection method of [18] is modified so that the boundary becomes a broad boundary, and a set of constraints on consistent combinations of the 9-intersection are developed. This results in 44 relations. Analysis reveals that the two calculi are identical except that SDMOBB coalesces cases 30 and 31 of the egg-yolk calclulus since the meeting of the broad boundaries in case 30 is regarded as an intersection just as is the case in 31 where there is a proper overlap. Similarly, cases 37 and 38 are coalesced in SDMOBB.

For any given configuration in the egg-yolk calculus one can consider the possible ways in which each region can be consistently "crisped"; depending on exactly how the crisp regions are formed (i. e. depending on exactly where the boundary between yolk and egg is drawn), different RCC-5 relations may pertain. However not all RCC-5 relations are possible for any given configuration. There are 13 equivalence classes formed from the 46 relations under this procedure, and these can be regarded as forming a coarser grained calculus for regions with indeterminate boundaries.

A similar clustering into groups of relations has also been achieved for the SDMOBB calclulus, in this case into 14 clusters, by considering the possible geometric interpretations of each relation and clustering similar ones together. Four relations are excluded from this analysis in order to simply the graph – those cases where the broad boundary has to be large enough to encompass the other region entirely (relations 19, 28, 34, 42 using the egg-yolk numbering).

Conceptual neighbourhoods have been defined for RCC-5, RCC-8, the 9-intersection calculi and indeed many other qualitative spatial calculi, which specify which relations are conceptually "close"; this conceptual closeness is usually defined in the sense that continuous deformation or movement of the regions involved results in the change of relation to one of the immediately conceptually neighbouring ones. This notion can also be defined for the egg-yolk calclulus and for SDMOBB.

The egg-yolk analysis can also be extended so that RCC-8 rather than RCC-5 is used as the underlying calculus. It turns out that if one generalizes RCC-8 in this way [16] there are 252 JEPD relations between non crisp regions which can be naturally clustered into 40 equivalence classes. An axiomatic presentation of the theory is also presented here which relies on an additional binary primitive relation to RCC (as well as the original $\mathbf{C}(x,y)$ relation) – x is crisper than region y. A mereolological framework which encompasses both standard RCC and the egg-yolk calculus in a semantic setting has also been developed [28].

It has been shown [12] that SDMOBB can be used to reason not just about regions with indeterminate boundaries but also can be specialized to cover a number of other kinds of regions including convex hulls of regions, minimum bounding rectangles, buffer zones and rasters. (This last specialization generalizes the application of the n-intersection model to rasters previously undertaken [20].)

It is also noted that the calculus can be used for reasoning for regions with holes (by reinterpreting the "yolk" as a hole). A similar calculus for lines with broad boundaries has also been developed [10]. As an alternative to using the 9-intersection method, the "calculus based method" for defining topological relations [13] has also been applied to defining SDMOBB [14].

Another notion of indefiniteness relates to locations. Bittner [4] deals with the notion of exact, part and rough location for spatial objects. The exact location is the region of space taken up by the object. The notion of part location (as introduced by [8]) relates parts of a spatial object to parts of spatial regions. The rough location of a spatial object is characterized by the part location of spatial objects with respect to a set of regions of space that form regional partitions. Consequently, the notion of rough location links parts of spatial objects to parts of partition regions.
Bittner [4] argues that the observations and measurements of location in physical reality yield knowledge about rough location: a vaguely defined object *o* is located within a regional partition consisting of the three concentric regions: 'core', 'wide boundary' and 'exterior'. In this context, the notion of rough location within a partition consisting of the three concentric regions coincides with the notion of vague regions introduced by [15].

Key Applications

Of all application areas for reasoning about space, geography abounds in examples of objects whose precise extent or location is hard to pin down precisely; from hills/valleys, to meterological entities and habitats, few geographic entities can be precisely spatially located and/or bounded. Perhaps only political entities, with *fiat boundaries* [9] are precisely defined (even if sometimes in dispute!). The potential for applying theories which handle spatial indeterminacy is broad.

Future Directions

It is worth noting the similarity of these ideas to rough sets [17], though the exact relationship has yet to be fully explored, though see, for example [5,24]. Other approaches to spatial vagueness are through the use of supervaluation theory [2,3,22,26], the use of fuzzy calculi [7], to work with an indistinguishability relation which is not transitive and thus fails to generate equivalence classes [21,29] and the development of nonmonotonic spatial logics [1,27].
The issue of whether the egg-yolk calclulus (and equivalently SDMOBB) really capture the notion of spatial *vagueness* properly with a tripartite division of space has been discussed [16] and an argument presented that the egg-yolk calculus can provide a way of interpreting a more general, higher order, axiomatization of vagueness.

Cross References

► Conceptual Neighborhood
► Mereotopology

Recommended Reading

1. Asher, N., Lang, J.: When nonmonotonicity comes from distance. In: Nebel, B., Dreschler-Fischer, L. (eds.): KI-94: Advances in Artificial Intelligence. pp. 308–318. Springer-Verlag, Heidelberg (1994)
2. Bennett, B.: Application of supervaluation semantics to vaguely defined spatial concepts. In: Montello, D.R. (ed.) Spatial Information Theory: Foundations of Geographic Information Science; Proceedings of COSIT'01, volume 2205 of LNCS, pp. 108–123. Springer, Morro Bay (2001)
3. Bennett, B.: What is a forest? on the vagueness of certain geographic concepts. Topoi. **20**(2), 189–201 (2001)
4. Bittner, T.: On ontology and epistemology of rough location. In: Freksa, C., Mark, D.M. (eds.) Spatial Information Theory – Cognitive and Computational Foundations of Geographic Information Science, number 1661 in Lecture Notes in Computer Science, pp. 433–448. Springer, Proceedings of COSIT'99 (1999)
5. Bittner, T., Stell, J.G.: Rough sets in qualitative spatial reasoning. In: Rough Sets and Current Trends in Computing, Banff, October 2000, Proceedings, volume 2005 of Lecture Notes in Computer Science, pp. 445–453. Springer, Heidelberg (2001)
6. Bittner, T., Stell, J.G.: Stratified rough sets and vagueness. In: Kuhn, W., Worboys, M.F., Timpf, S. (eds.) Spatial Information Theory. COSIT'03 Proceedings, volume 2825 of Lecture Notes in Computer Science, pp. 286–303 (2003)
7. Burrough, P., Frank, A.M. (eds.): Specialist Meeting on Geographical Objects with Undetermined Boundaries, GISDATA. Francis Taylor (1996)
8. Casati, R., Varzi, A.: The structure of spatial localization. Philos. Stud. **82**, 205–239 (1996)
9. Casati, R., Varzi A.: Fiat objects. Topoi, **20**,131–148 (2001)
10. Clementini, E.: A model for lines with a broad boundary. In: 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU 2002),Annecy, France, number 692 in LNCS, pp. 1579–1586 (2002)
11. Clementini, E., Di Felice, P.: An algebraic model for spatial objects with undetermined boundaries. In: Burrough, P., Frank, A.M. (eds.): Proceedings, GISDATA Specialist Meeting on Geographical Entities with Undetermined Boundaries. Taylor Francis, London (1996)
12. Clementini, E., Di Felice, P.: Approximate topological relations. International Journal of Approximate Reasoning, **16**,173–204, (1997)
13. Clementini, E., Di Felice, P., Oosterom, P.: A small set of formal topological relationships suitable for end user interaction. In: Abel, D., Ooi, B.C. (eds.): Advances in Spatial Databases, Proceedings of the 3rd International Symposium on Spatial Databases (SSD'93), Singapore, number 692 in LNCS, pp. 277–295. Springer-Verlag, Heidelberg (1994)
14. Clementini, E., Di Felice, P.: A spatial model for complex objects with a broad boundary supporting queries on uncertain data. Data Knowl. Eng. **37**(3), 285–305, (2001)
15. Cohn, A.G., Gotts, N.M.: The 'egg-yolk' representation of regions with indeterminate boundaries. In: Burrough, P., Frank, A.M. (eds.): Proceedings, GISDATA Specialist Meeting on Geographical Objects with Undetermined Boundaries, pp. 171–187. Taylor Francis, London (1996)
16. Cohn, A.G., Gotts, N.M.: A mereological approach to representing spatial vagueness. In: Doyle, J. Aiello, L.C., Shapiro, S. (eds.): Principles of Knowledge Representation and Reason-

ing, Proc. 5th Conference, pp. 230–241. Morgan Kaufmann, San Mateo (1996)
17. Düntsch, I., Gediga, G.: Uncertainty measures of rough set predictions. Artif. Intell. **106**, 109–137 (1998)
18. Egenhofer, M.: Deriving the composition of binary topological relations. J. Vis. Lang. Comput. **5**(2), 133–149 (1994)
19. Egenhofer, M., Franzosa, R.: Point-set topological spatial relations. Int. J. of Geogr. Inf. Syst. **5**(2), 161–174 (1991)
20. Egenhofer, M.J., Sharma, J.: Topological relations between regions in R^2 and Z^2. In: David, A., Ooi, B. (eds.): Proceedings of the 3rd International Symposium on Advances in Spatial Databases, SSD'93, volume 692 of LNCS, Springer, Heidelberg (1993)
21. Kaufman, S.: A formal theory of spatial reasoning. In: Proc Int. Conf. on Knowledge Representation and Reasoning, pp. 347–356, Morgan Kaufmann, San Mateo (1991)
22. Kulik, L.: A geometric theory of vague boundaries based on supervaluation. In: Montello, D.R. (ed.): Spatial Information Theory: Foundations of Geographic Information Science, International Conference, COSIT 2001, Morro Bay, CA, USA, September 19–23, 2001, Proceedings, volume 2205 of Lecture Notes in Computer Science, pp. 44–59. Springer, Heidelberg (2001)
23. Lehmann, F., Cohn, A.G.: The EGG/YOLK reliability hierarchy: Semantic data integration using sorts with prototypes. In: Proc. Conf. on Information Knowledge Management, pp. 272–279. ACM Press, New York (1994)
24. Polkowski, L., Skowron, A.: Rough mereology in information systems. a case study: qualitative spatial reasoning. In: Rough set methods and applications: new developments in knowledge discovery in information systems, pp. 89–135. Physica-Verlag GmbH, Heidelberg, Germany, (2000)
25. Randell, D.A., Cui, Z., Cohn, A.G.: A spatial logic based on regions and connection. In: Proc. 3rd Int. Conf. on Knowledge Representation and Reasoning, pp. 165–176. Morgan Kaufmann, San Mateo (1992)
26. Santos, P., Bennett, B., Sakellariou, G.: Supervaluation semantics for an inland water feature ontology. In: Kaelbling, L.P., Saffiotti, A. (eds.): Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI-05), pp. 564–569. Professional Book Center, Denver (2005)
27. Shanahan, M.: Default reasoning about spatial occupancy. Artif. Intell. **74**(1),147–163 (1995)
28. Stell, J.G.: Part and complement: Fundamental concepts in spatial relations. Ann. Artif. Intell. Mat. **41**,1–18 (2004)
29. Topaloglou, T.: First order theories of approximate space. In: Anger, F. et al. (ed.): Working notes of AAAI workshop on spatial and temporal reasoning, Seattle, pp. 283–296 (1994)

Resolution-Based Matrix Quadtree

► Geospatial Authorizations, Efficient Enforcement

Resolving Semantic Schema Heterogeneity

► Database Schema Integration

Resource Description Framework (RDF)

► Geography Markup Language (GML)

Retrieval Algorithms, Spatial

MARK SANDERSON
Department of Information Studies,
University of Sheffield, Sheffield, United Kingdom

Synonyms

Geographic information retrieval; Information retrieval; Toponym; Footprint; Synonymy; Ambiguity; Gazeteer; Triage, query; Relevance, spatial; Relevance, textual

Definition

Before describing means of adapting text search engines to deal with location, it is important to both define what is meant by spatial retrieval and further to establish how often users actually need access to search systems which provide a spatial component. Purves and Jones define spatial retrieval as [16]

"...the provision of facilities to retrieve and relevance rank documents or other resources from an unstructured or partially structured collection on the basis of queries specifying both theme and geographic scope."

Historical Background

In determining user needs for such location-based search, one can start by examining the range of existing search engines with explicit support for spatial search. Probably the most obvious starting point for examples of such systems is to examine the major Web search engines, such as Google, Yahoo or MSN. All provide so-called *local search* facilities, which essentially provide spatial search over yellow pages (i. e. directories of businesses). The sophistication of such searching is limited to querying for the name or type of businesses typically found in yellow pages and specifying a spatial footprint, using a postal code or the name of a population center. The prominence of local search on all three of the main engines indicates its prominence. It is also worth noting that search engines – when showing advertising next to their standard search results – try to provide adverts for goods and services located close to the queryier's actual location. This they do by examining the IP address of the searcher's computer, which can often be used to determine the location of the user at least to the spatial resolution of the country the searcher's computer is in [5].

For more general search of web pages, it would appear that the search engines do not provide any explicit support for spatial search[1]. In analyses of query logs, it was found that around 20% of searches included a location within them [13]. The analyses examined the range of spatial relations that users specified in their searches, of which almost all used "in", others such as "near", "north of", "surrounding", etc. were much less common, (occurring one or two orders of magnitude less frequently). A simple word based search for a particular topic (e. g. battles) *in* a particular location (e. g. some region of a country) is well served by the search engines' usual requirement that matching documents contain all words of the users' query. Even though other spatial relations aren't as well served by conventional text search, because they aren't used as much, it would appear search engines aren't motivated to alter their engines to support the relations.

From such an examination of services provided and the queries submitted to search engines, it would appear that users conduct spatial search relatively often, but it would also appear that, in general, insufficient numbers of users require support for complex spatial relations to make is worthwhile providing.

Spatial Search Needed on the Web?

A further reason that Web search engines may not be providing specific support for spatial search is that even when users search for objects with a particular spatial region that isn't mentioned on a particular web page, often such pages are still easy to find. There are a great many Web pages that simply list items found within a particular spatial area and provide pointers to the page describing those items in more detail. A series of searches on any Web search engine will confirm the presence of such lists: Castles in Scotland, hotels in New England, for example. Even if the Web site of a hotel in downtown Boston fails to mention that the hotel is in New England, the page will still be findable via a list page. Even searches for apparently obscure items such old yew trees in England or Vitrified Forts in Yorkshire often produce relevant useful list pages, which satisfy most searchers. (Academic research on geographic text search has confirmed the success of simple keyword search when searching for locations [6]. With this combination of straightforward user queries coupled with large sets of list pages grouping items spatially, it is perhaps unsurprising that search engines appear to do little to further support spatial search.

[1]It should be noted that the exact means by which Web search engines provide their services are closely guarded secrets and the methods used by them change continuously. Therefore, any statement about how such search engines work (from any source, not just this text) should be taken to be little more that informed speculation that might go out of date at any time.

Key Applications

It would be wrong to assume that the services offered by the major Web search engines in some way cover or define all possible searching needs. Users searching for other types of objects or searching in other domains can have quite different priorities. One example of this is searching for photographs via text captions. Here, location appears to be of great importance. An examination of requests made by journalists to photo libraries [3], and more recently of query logs maintained by image digital libraries [12] revealed that between a 1/3 and 1/2 of all requests to such libraries were related to a specific place, region or country; or the request was for a certain form of landscape (e. g. river, sea, countryside, mountains); or a particular permanent man made feature that would typically be represented in a GIS, (e. g. railway station, school, park, church, etc.). An examination of the keywords manually assigned to images in Flickr reveals a similar emphasis on location.

There would also appear to be an increase in the number of cameras produced that provide some form of location information either through GPS or on camera phones that determine location using data from the cell phone operators [1]. From such devices one might be able to automatically caption photographs and research is on-going to examine the potential of this approach [11,7].

Specialized Spatial Search Services

It would also appear that there are specialized search problems that require consideration of spatial search. For such problems, companies exist to provide solutions. At the time of writing probably the best example of such a company is MetaCarta, which provides spatial search tools for such needs.

From this overview, it is clear that spatial search is a common though not dominant form of search and that different forms of searching require different types of spatial search. The means one can use to support such search are next described.

Scientific Fundamentals

Spatial search requires different components in order for it to be supported well. First, it may be necessary to determine if users are issuing a spatial search, if they are, then both the user's query and each of the documents in the collection to be searched need to have some form of location associated with them. When a search to such a system is issued, those documents that are both semantically and

spatially relevant to the user's query need to be found as quickly as possible. This Section describes these stages of spatial search, starting with a brief overview of the basic task of determining if a text contains a reference to a location (formally known as a *toponym*) within it and establishing what area the location is referring to (known as a *footprint*).

Identifying Toponyms in Text

A detailed exploration of toponym identification and resolution is beyond the scope of this Section, however, accurately spotting all toponyms and resolving which location is being referred to is not as straightforward as one might initially imagine. As with language in general, there are two basic problems in identifying toponyms and resolving (*grounding*) the toponyms to a location: *synonymy* (different ways of referring to the same thing) and *ambiguity* (a word having different meanings). Each is briefly described here.

Synonymy

There are many different ways in which a location can be referred to in a text and these ways can be both explicit, implicit and vague. The commonest explicit way of referring to a location in text is simply to provide the name of a location. Identifying such a location in a text can simply be achieved with the use of a large place name gazetteer, of which both open source and licensed versions can be found. A full address in a text is another common example of an explicit location and again gazetteers containing lists of addresses have been created.

One can also implicitly determine location by examining items such as telephone numbers. As with names of places and addresses, gazetteers that list what location corresponds to a particular phone number exist. There can also be domain specific implicit references to location.

- Wang showed that many user Web queries can be implicitly spatial [17]: one word queries for services, such as cinemas or pizza restaurants are likely to be best served by a spatial search if the search engines has access, through some other means, to the location of the user issuing the query.
- If a text comes to a computer associated with an IP address (for example query text transmitted to a remote search engine), most IP addresses can be resolved to an accuracy of at least a country, often to a city and sometimes to very precise locations (e. g. if an IP address comes from a free wireless service at an Internet café).
- If one is trying to associate a location with a set of Web pages, one might identify a location on one page, e. g. the "Contact Us" page of a business Web site and the assume other pages from the same Web site have the same location associated with them.

An often overlooked form of reference to locations are *imprecise regions*: these are vague locations commonly used whose boundaries aren't precisely defined. Examples of such areas are

- the "Midlands" of England a large area centered on the city of Birmingham;
- the "Eastern Seaboard" of the United States, a long region of the US east coast encompassing cities such as Boston, New York, Washington, etc., but whose western boundaries aren't as clearly defined;
- the "Red Centre" of Australia, the largely un-inhabited central dessert region of the country.

Resources to help determine the location and shape of such regions are sparse, despite use of such regions being relatively common in both documents and in user queries. Research on automatically establishing the shape of imprecise regions using text mining approaches has been reported [2].

Ambiguity

In general, words are ambiguous, with many words having more than one meaning: the classic example of such ambiguity is the word "bank", which can refer to both an economic institution and the side of river, as well as other less well known meanings. Unfortunately, such ambiguity extends to toponyms as well. This has been studied by Smith and Mann [14], who showed that particularly in countries colonized relatively recently, such as those in North and Central America, many place names are re-used. For example, there is a "Springfield" in almost every state of the US. Unfortunately, place names are not necessarily just spatially ambiguous, they can also be ambiguous with other types of words. They could also be:

- names of people (e. g. the state of "Victoria" in Australia; the city of "Washington" in the US, etc.);
- used by organizations taking on the name of their location (e. g. football clubs, such as "Manchester United", "Real Madrid", "AC Milan", etc.);
- used metaphorically (e. g. one might use the term "Hollywood" to refer to the US film industry rather than the place);
- used in different languages having quite different meanings (e. g. there is a place in Germany called "the");
- used to mean different things (e. g. a 5 digit US postal code might simply be used in a text as a number).

Accurately identifying words as toponyms has been studied for many years with research conducted as part of the Message Understanding Conferences (MUC) of the late 1990s. More recently, work on toponym resolu-

tion has increased. Research and an extensive literature review from Leidner showed that fully automated accurate toponym resolution is hard to achieve and is still an active area of research [9].

Query Triage

Depending on the form of search being conducted, it may be necessary to automatically determine if a user's query is a spatial one or not. Examples of systems that need to make such a determination are Web search engines, which by default search for web pages, but may need to direct a spatial query to a local search service. Some work in this area of so-called *query triage* has been published [17]. In most cases, simply examining the user's query for the presence of a place name (taken from a large gazetteer) is sufficient for identifying a potentially spatial query.

Once a spatial search has been identified, the next stage is to establish the components of the query: the subject of the query, the location the query pertains to and possibly, the spatial relation between the subject and the query. Recognizing the spatial relation in a query text is a relatively un-explored aspect of research, however a simple search of words commonly used to define a spatial relation (e. g. "in", "north of", "near", "close to", etc.) is likely to suffice most times.

A much simpler way of determining the components of a spatial query is to design the user interface of a search system to capture these elements individually.

Conducting a Spatial Search

Assuming one has received a query with an identified location and relation that will search a collection of documents which themselves each have a location associated with them, one needs to consider how best to retrieve the documents. When matching documents to a spatial query, there are two aspects to be considered: the textual relevance of a document to the query as well as the document's spatial relevance.

Textual Relevance

Details on the means to determine textual relevance is beyond the scope of this document. The exact methods used to calculate this very much depend on the type of documents being searched: for document collections, such as office files, newspaper articles or email finding, documents holding the user's query words repeated many times within the document is a simple but effective approach (see [4] for more coverage on this topic). If searching Web documents, consideration of the number of links pointing to a page and the *anchor text* written in those links takes priority over query word density (see [10] for more details).

Spatial Relevance

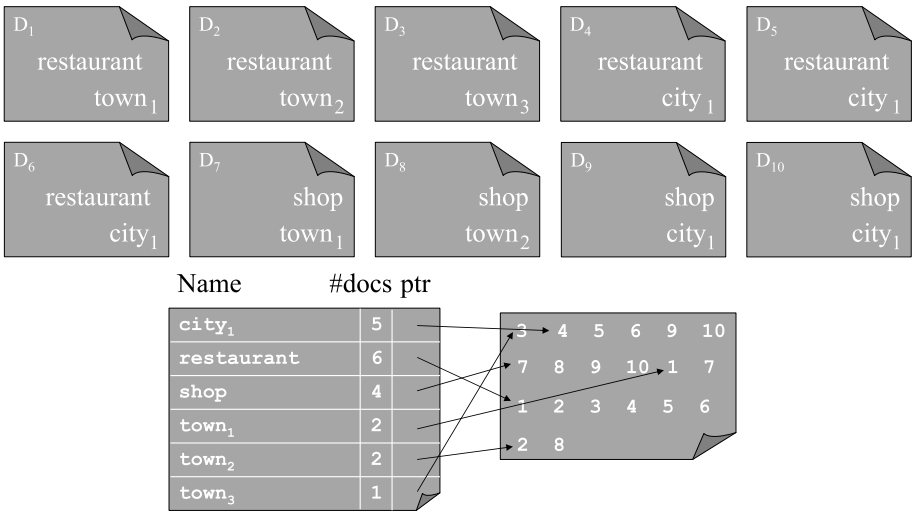
Determining the spatial similarity of a document to a query starts by establishing the spatial footprint of the query, followed by selecting documents that are relevant to the spatial area of the query. The footprint of a query is determined by the location and any spatial relation specified in the query. Determining the size and shape of a footprint is dependent on the subject of the query: if users are searching for restaurants in a city, they are likely to be implying that the eateries must be found within the city limits; however users searching for an airport in a city, may be happy with matches well beyond the city's border.

Once a footprint has been established, documents found to be both textually and spatially relevant are retrieved. Means of combining the scores of the spatial and textual components were described by van Kreveld et al. [8]. One important aspect to searching spatially is ensuring that the documents returned to users are not just those spatially relevant a particular sub-area of the query footprint, but are instead dispersed as evenly as possible around the footprint area. The van Kreveld paper described algorithms for achieving this dispersion searching and reported improved user satisfaction with such a searching system.

Internals of Search

Speed of search is a crucially important aspect of search. Studies have found that users use search engines less frequently if the speed of response of the engine is anything more than a 1/4 of a second. Ensuring that search is completed a quickly as possible is critical to building a successful searching system. In this Section an overview of the basic method used by ordinary text search engines to provide fast search is described, followed by a description of the means used to adapt a search engine to serve spatial queries.

To achieve fast search over a large collection of documents, a standard search engine uses a combination of a sorted word *index* and an *inverted file* (sometimes referred to as a *postings file*) to facilitate fast search of documents matching a particular query. Figure 1 shows a simplified example of such a combination of files, built from a collection of ten "toy" documents, which describe a set of shops and restaurants in three towns (named $town_1$, $town_2$, $town_3$ and one city (named $city_1$). Each document is identified by an ID number ranging from 1–10. The index holds a sorted list of all the words contained in all the documents. Each entry in the index has the word, the number of documents that word occurs in, and a pointer to the place in the inverted file where the ids of the documents holding that word are located. For a detailed outline of how such files are created and means of refining the structures shown here, see [18].

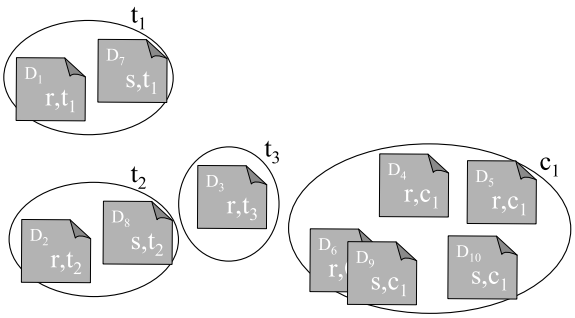


Retrieval Algorithms, Spatial, Figure 1 'Toy' collection and a conventional word-based index of the collection

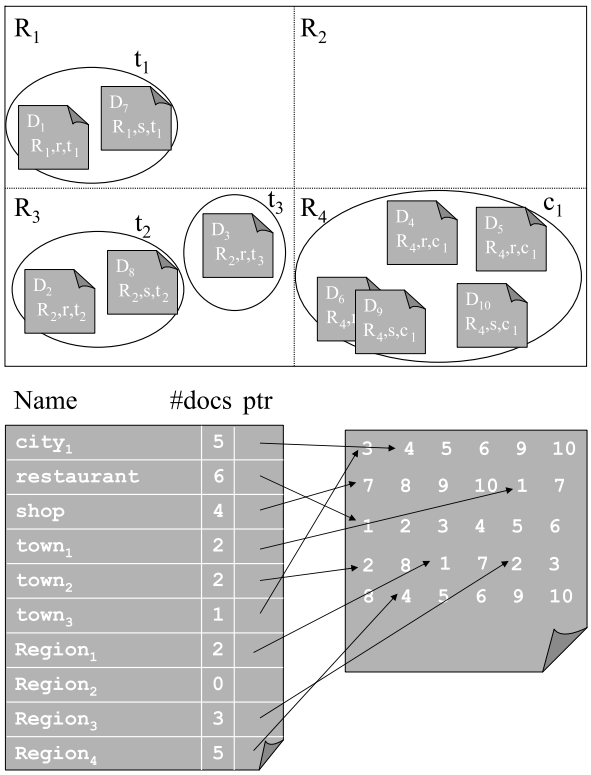
Adapting Text Search to Spatial Search

Such an index already supports simple spatial search: users wishing to find a restaurant in $town_1$ simply type the query "restaurant $town_1$"; Taking the intersection of the list of document IDs containing the word restaurant and the list containing the word $town_1$ results in a single document, which is shown to the user. If users wished to find restaurants near to and in $town_1$, it would be necessary to know the spatial location of each town and city. With such information (see Fig. 2), if it was judged that $town_2$ and $town_3$ were near to $town_1$, a spatial search system could simply issue three separate queries "restaurant $town_1$", "restaurant $town_2$", "restaurant $town_3$" and show the union of the returned results.

This simplistic approach of conducting multiple searches, one for each location that is spatially relevant to a query is a reasonable method if the number of named locations is relatively small, as is the case with the example collection. If the collection is much larger, however, issuing multiple queries starts to be time consuming. Borrowing methods



Retrieval Algorithms, Spatial, Figure 2 'Toy' collection spatially arranged

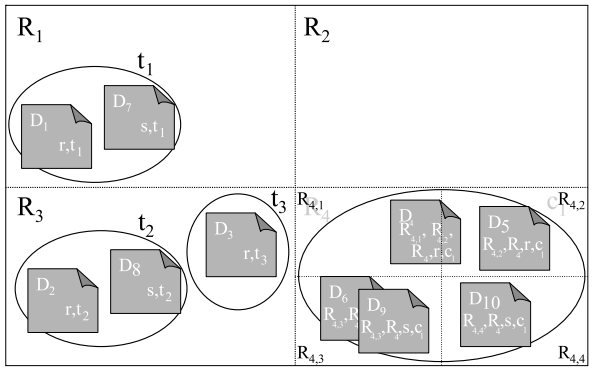


Retrieval Algorithms, Spatial, Figure 3 Organising documents into regions, adding region keywords to the index

from spatial indexing, the map in Fig. 3 can be broken into regions (R_1, R_2, R_3, R_4) and keywords[2] associated with each of these regions can be assigned to the relevant documents (see Fig. 4).

[2]Care needs to be taken to ensure that such keywords do not clash with actual words found in documents.

R



Retrieval Algorithms, Spatial, Figure 4 Further sub-dividing one region into smaller spatial areas

If a particular area, such as the city, isn't spatially represented in sufficient detail, its region, R_4, can be further sub-divided ($R_{4,1}$, $R_{4,2}$, $R_{4,3}$, $R_{4,4}$) and keywords associated with each region additionally assigned to the five documents in that region. If a document is found to be spatially relevant to more than one region, it can be assigned multiple region keywords. With such a representation in place, queries for shops in the south west of the city can be issued with a manageable number of searches, as can queries for restaurants near to $town_1$. Other types of data structure for providing fast spatial search have been examined and are described in [15], which also contains an extensive review of other spatial search research.

Future Directions

With the rapid uptake of geo-location devices, such as GPS, being placed in more mobile devices, such as cell phones and digital cameras, it is reasonable to expect that search over data that has a location associated with it will increase and the consequent important of spatial retrieval algorithms will grow.

Cross References

► Indexing Spatio-temporal Archives
► Representing Regions with Indeterminate Boundaries

Recommended Reading

1. Ahern, S., Davis, M., King, S., Naaman, M., Nair, R.: Reliable, User-Contributed GSM Cell-Tower Positioning Using Context-Aware Photos. In: The Eighth International Conference on Ubiquitous Computing (UbiComp) (2006)
2. Arampatzis, A., van Kreveld, M., Reinbacher, I., Jones, C.B., Vaid, S., Clough, P.D., Joho, H., Sanderson, M.: Web-based delineation of imprecise regions. J. Comput. Environ. Urban Syst. (CEUS) **30**(4) (2006)
3. Armitage, L.H., Enser, P.G.B.: Analysis of user need in image archives. J. Inf. Sci. **23**(4): 287–299 (1997)
4. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval, Addison Wesley, (1999) ISBN-13: 978–0201398298
5. Buyukkokten, O., Cho, J., Garcia-Molina, H., Gravano, L., Shivakumar, N.: Exploiting geographical location information of web pages. In: Proceedings of Workshop on Web Databases (WebDB'99) held in conjunction with ACM SIGMOD'99 (1999)
6. Gey, F., Larson, R., Sanderson, M., Joho, H., Clough, P.: GeoCLEF: the CLEF 2005 Cross-Language Geographic Information Retrieval Track. In: The Working Notes for the CLEF 2005 Workshop (2005)
7. Iwasaki, K., Yamazawa, K., Yokoya, N.: An Indexing System for Photos Based on Shooting Position and Orientation with Geographic Database. In: Proceedings of IEEE international conference on multimedia and expo (ICME), 390–393 (2005)
8. van Kreveld, M., Reinbacher, I., Arampatzis, A., van Zwol, R.: Distributed Ranking Methods for Geographic Information Retrieval. In: Developments in Spatial Data Handling, pp. 231–243. Springer, Berlin, Heidelberg (2004)
9. Leidner, J.L.: Toponym Resolution in Text: Annotation, Evaluation and Applications of Spatial Grounding of Place Names. PhD thesis, School of Informatics, University of Edinburgh, Edinburgh, Scotland, UK (2007)
10. Levene, M.: An Introduction to Search Engines and Web Navigation. Pearson, (2005) ISBN13: 9780321306777
11. Naaman, M., Song, Y.J., Paepcke A., Garcia-Molina H.: Assigning Textual Names to Sets of Geographic Coordinates. J. Comput. Environ. Urban Syst. J. **30**(4):418–435 (2006)
12. Paterson, C.: The effectiveness of using machine translators to translate German and Portuguese into English when searching for images with English captions. Masters Dissertation, Department of Information Studies, University of Sheffield (2002)
13. Sanderson, M., Kohler, J.: Analyzing geographic queries. In: The proceedings of Workshop on Geographic Information Retrieval SIGIR (2004)
14. Smith, D.A., Mann G.S.: Bootstrapping toponym classifiers. In: Proceedings of the Workshop on the Analysis of Geographic References, at the NAACL 2003 Conference (2003)
15. Purves, R.S., Clough, P., Jones, C.B., Arampatzis, A., Bucher, B., Finch, D., Fu, G., Joho, H., Syed, A.K. Vaid, S., Yang, B.: The design and implementation of SPIRIT: a spatially aware search engine for information retrieval on the Internet. Int. J. Geogr. Inf. Sci. (2007)
16. Purves, R.S., Jones, C.B.: Geographic Information Retrieval (GIR) editorial. In: special issue Comput. Environ. Urban Syst. **30**(4): 375–377 (2006)
17. Wang, L., Wang, C., Xie, X., Forman, J., Lu, Y., Ma, W., Li, Y.: Detecting Dominant Locations from Search Queries. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2005) (2005)
18. Zobel, J., Moffat, A.: Inverted Files for Text Search Engines. ACM Computing Surveys, **38**(2): 1–56 (2006)

Reuse

► Smallworld Software Suite

Reverse-K-Nearest-Neighbors aNN

► Nearest Neighbors Problem

Reverse-Nearest-Neighbor-Problem

► Nearest Neighbors Problem

Reversible and Convertible Lanes

► Contraflow for Evacuation Traffic Management

Revisit Period

► Temporal Resolution

RF Identification

► Radio Frequency Identification (RFID)

RFID

► Radio Frequency Identification (RFID)

Rich Client Internet Applications

► Scalable Vector Graphics (SVG)

rkNN

► Nearest Neighbors Problem

RMS Error

► Photogrammetric Products

rNN

► Nearest Neighbors Problem

Road Maps

► Road Maps, Digital

Road Maps, Digital

BETSY GEORGE, SHASHI SHEKHAR
Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA

Synonyms

Digital road networks; Road maps

Definition

A digital road map is a representation of a physical road network that can be displayed or analyzed by a digital computer. Figure 1 shows a road map and its graph representation. Road intersections are often modeled as vertices and the road segments are connecting adjacent intersections represented as edges in the graph. For example, the intersection of 'SE 5th Ave' and 'SE University Ave' is modeled as node N1. The segment of 'SE 5th Ave' between 'SE University Ave' and 'SE 4th Street' is represented by the edge N1-N4. The directions on the edges indicate the permitted traffic directions on the road segments.

Digital road maps have gained importance due to the widespread use of location-based services such as route finding. They are essential in any location-based utility that involves route-based queries.

Historical Background

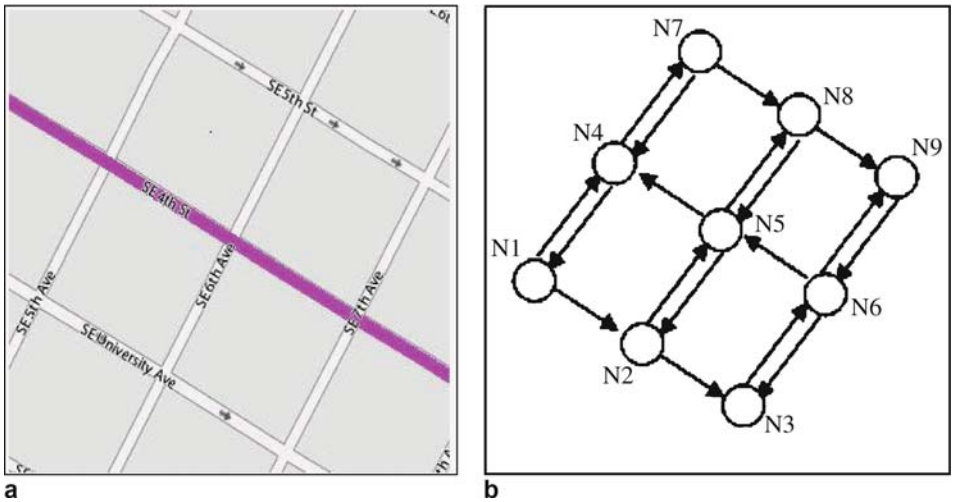
Traditionally, governmental agencies (e. g. state departments of transportation) were the primary sources of digital road maps. The initial efforts to formulate digital maps were undertaken by governmental agencies (e. g. state departments of transportation) through the digitization of paper maps from various sources. TIGER (Topologically Integrated Geographic Encoding and Referencing system) was conceived and developed in the 1980's in preparation for the 1990 Census by the US Census Bureau. Due to an increased popular demand, several private-sector companies have begun to offer digital road maps appended with additional points of interest and improve map accuracy.

Scientific Fundamentals

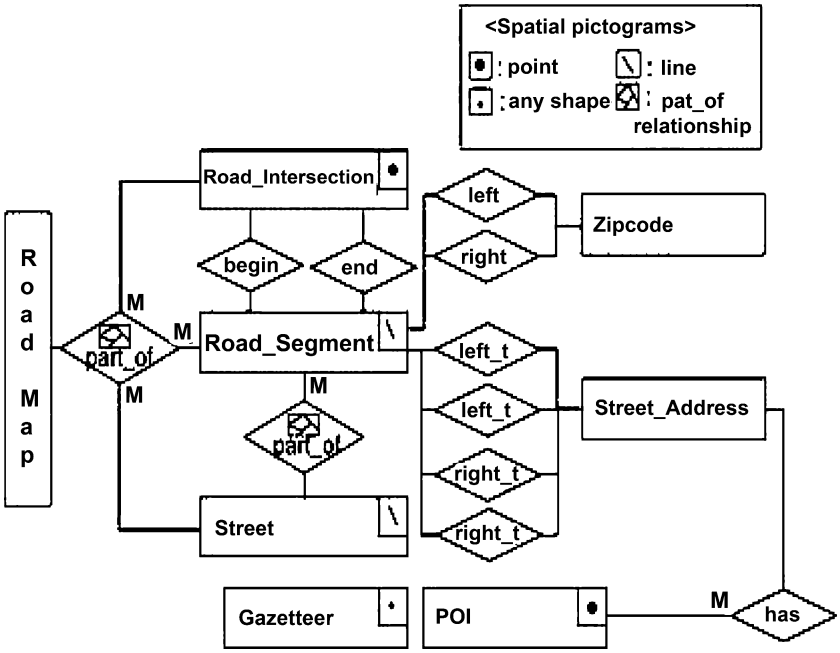
Data Model of Digital Road Maps

As in any database application, data modeling of digital road maps is critically important. The database design involves three steps namely conceptual modeling, logical modeling and physical modeling [2].

Conceptual Data Model The purpose of conceptual modeling is to adequately represent the data types, their



Road Maps, Digital, Figure 1 Road map and its graph representation (Source for Figure 1a: www.maps.yahoo.com)



Road Maps, Digital, Figure 2 A PEER diagram for a digital road map [1]

relationships and the associated constraints. Entity Relationship (ER) model, that has been widely used in conceptual modeling does not offer adequate features to capture the spatial semantics of road maps. Several extensions to address this limitation have been proposed. The pictogram enhanced ER (PEER) model [3] models road maps as graphs. Figure 2 shows a PEER diagram for a digital road map. In a digital road map, vertices represent road intersections and edges represent road segments. Labels and weights can be attached to vertices and edges to encode additional information such as names and travel times. A road segment is modeled using a range of street addresses, which are divided into left-side and right-side addresses. Each side of the road segment is represented using the two end-addresses. The zip code information of a street address can be used when the exact address is not known. Zip codes corresponding to the left and right sides of the streets are also stored. Two edges are considered to be adjacent if they share a common vertex. A PEER diagram can also represent points of interest.

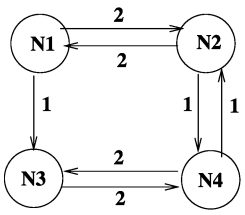
Logical Data Model In the logical modeling phase, the conceptual data model is implemented using a commercial database management system. Among the various implementation models such as hierarchical, network, relational, object-relational data models and object-oriented models, the object-relational model has been gaining popu-

larity in the representation of spatial applications. This model is supported by SQL3 and provides a mechanism for user-defined data types, thus allowing the definition of user defined complex data types such as point, line and polygon. [3] provides the grammar-based translation scheme to translate the PEER model into an object relational (OGIS/SQL3) model. In general, entity pictograms translate into appropriate data types in SQL3 and the relationship pictograms translate into spatial integrity constraints [1].

Physical Data Model The physical data modeling phase deals with the actual implementation of the database application. Issues related to storage, indexing and memory management are addressed in this phase. Very often, queries that are posed on a network database such as a road map, involve route finding. This means the database must provide adequate support for network computations such as finding shortest paths. Figure 3 shows three representations of a graph. Adjacency-matrix and adjacency list are two well-known data structures used for implementing road networks [4], represented as graphs. In an adjacency-matrix, the rows and columns of a matrix represent the vertices of the graph. A matrix entry can be either 1 or 0, depending on whether there is an edge between the two vertices as shown in Fig. 3b. An adjacency list (shown in Fig. 3c) consists of an array of pointers. Each element of the array represents a vertex in the graph and the pointer points to a list of vertices that are adjacent to the vertex. Directed graphs can be implemented in the relational model using a pair of relations, one for the nodes and the other for the edges. The 'Node' (R) and the 'Edge' (S) relations are shown in Fig. 3d and a denormalized representation is shown in Fig. 3e. The denormalized representation of a node table contains the coordinates of the node, a list of its successors and a list of its predecessors. This representation is often used in shortest path computations.

A spatial access method called the Connectivity-Clustered Access Method (CCAM) was proposed in [6], which clusters the vertices of the graph based on graph partitions, thus providing grouping of records into disk pages based on connectivity.

Turn Restrictions Turn restrictions are frequently encountered in road networks and they can affect the traversal in the network. A physical model that does not consider turn restrictions can lead to the computation of routes that are not entirely feasible. Turns have been modeled using a turn table where each turn restriction is represented as a row in the table that references the two associated edges [9]. Another proposed method to represent turn restrictions is node expansion [10]. The node that corresponds to a junction is expanded to a subgraph where permissible turns are represented as edges. This technique can lead to a substantial increase in the size of the network which adversely affects the performance. Another method involves the transformation of the road network to a line graph where the edges in the original network are mapped to vertices in the line graph and the turns are represented as edges in the line graph [9,11]. A representation that



a Example network graph

	N1	N2	N3	N4
N1	0	1	1	0
N2	1	0	0	1
N3	0	0	0	1
N4	0	1	1	0

b Adjacency matrix

N1	N2	N3
N2	N1	N4
N3	N4	
N4	N2	N3

c Adjacency list

Node (R)

ID	X	Y
N1	1	1
N2	3	1
N3	1	0
N4	3	0

Edge(S)

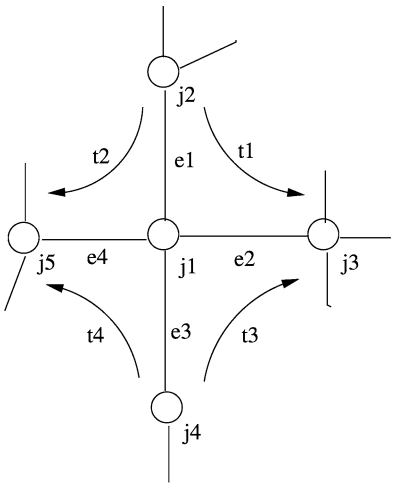
Src	Dst	Dist
N1	N2	2
N1	N3	1
N2	N1	2
N2	N4	1
N3	N4	2
N4	N2	1
N4	N3	2

d Node and Edge Relations

ID	X	Y	Succ	Pred
N1	1	1	(N2,N3)	N2
N2	3	1	(N1,N4)	(N1,N4)
N3	1	0	N4	(N2,N3)
N4	3	0	(N2,N3)	(N2,N3)

e Denormalized Node Table

Road Maps, Digital, Figure 3 Three different representations of a graph



a An example network

id	from–jn
e1	j2
e2	j3
e3	j4
e4	j5

b Edge Table

id	edge1	junc1	edge2	junc2	edge3	junc3	edge4	junc4
j1	e1	j2	e2	j3	e3	j4	e4	j5

c Junction Table

id	Turn id	First edge id	Last edge id	Turn id	First edge id	Last edge id	Turn id	First edge id	Last edge id	....
j1	t1	e1	e2	t2	e1	e4	t3	e3	e2	

d Turn Table

Road Maps, Digital, Figure 4 Representation of Turn Restrictions (adapted from [5])

consists of a junction table, edge table and turn tables was proposed in [5].

Every junction is represented as a row in the junction table. A row corresponding to a junction stores the edges that converge at the junction and the junctions connected to the given junction. The edge table stores edge identifiers and the junction where the edge originates (from-junction). A tuple in the turn table corresponds to a junction in the network. Each tuple consists of a junction identifier, and a triplet (turn identifier, first edge-id, last edge-id) corresponding to each turn associated with the given junction.

Figure 4 illustrates the representation of turn restrictions in a road network. Figure 4a shows a part of a road network around a junction *j1* where the edges *e1, e2, e3* and *e4* meet. The curved arrows indicate the permitted turns at the junction. For example, a turn is allowed from edge *e1* to edge *e*2. Figure 4b, c and d show the edge, junction and turn tables respectively, corresponding to turn *t1* in the example network. The junction table lists the edges that converge at junction *j1* (*e1, e2, e3,* and *e4* and the junctions connected to it (*j2, j3, j4,* and *j5*). The turn able shows the permitted turns at junction *j1* and the edges that participate in each turn. For example, turn *t1* represents a turn from edge *e1* to edge *e2* as illustrated by the *'first edge id'* and *'last edge id'* entries in the turn table in Fig. 4d.

Data Quality

Given the significant number of sources for the road map data and the heterogeneity across the data, it became necessary to define data quality in the context of digital road maps. Data quality refers to the relative accuracy and precision of a particular road map database. The purpose of the data quality report is to provide adequate information to the users to evaluate the fitness of the data for a specific use. There are several map accuracy standards, including the well-known National Map Accuracy Standard (NMAS) and the American Society for Photogrammetry and Remote Sensing (ASPRS) standard [1]. The standards consist of four components namely:

1. Lineage: This component deals with the narrative of the source materials used and procedures adopted to build the product.
2. Positional Accuracy: This defines the error in position of features. In digital road maps, this component is the most critical.
3. Attribute Accuracy: This represents the expected error in attributes such as road names.
4. Completeness: This defines the fraction of the real-world features represented on a map.

In addition, topological consistency is of concern for digital road maps in the context of navigation systems to facilitate graph computations such as shortest path algorithms.

Key Applications

Location-based Services

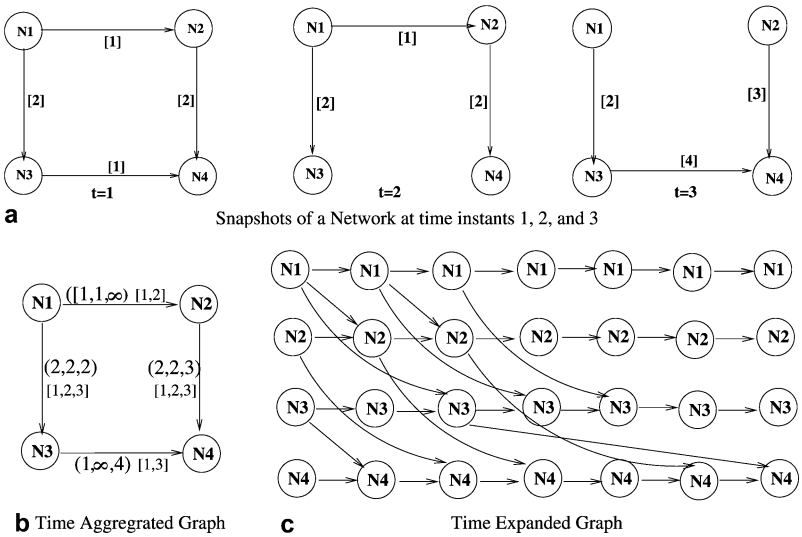
Digital road maps are indispensable for any location-based service that involves position or route based queries. Location-based services (LBS) provide the ability to find the geographical location of a mobile device and subsequently provide services based on that location. A digital road map is a key component of spatial database servers that provide efficient query-processing capabilities such as finding the nearest facility (e. g. a restaurant) and the shortest path to the destination from a given location [14,15]. Route-finding queries typically deal with route choice (shortest route to a given destination), destination choice (the nearest facility from the given location) and departure time choices (the time to start the journey to a destination so that the travel time is minimized). Though a significant amount of work has been done to find best routes and destinations, the problem of computing the best time to travel on a given route (time choice) needs further exploration.
Digital road maps are critical in in-vehicle navigations systems [12,13] where the maps would be used to compute the required routes on user-demand or to find points of interest.

Emergency Planning

One key step in emergency planning is to find routes in a road network to evacuate people from disaster-stricken areas to safe locations in the least possible time. This requires finding shortest routes from disaster areas to destinations. In metropolitan-sized transportation networks, manual computation of the required routes is almost impossible, making digital road maps an integral part in the efficient computation of these routes.

Future Directions

A significant fraction of queries that are posed on a road network involves finding the shortest path between a pair of locations. Travel times on the road segments, very often depend on the time of day due to varying levels of congestion, thus making the shortest paths also time-dependent. Road networks need to be modeled as spatio-temporal networks to account for this time-dependence. Various models such as time-expanded networks [7] and time-aggregated graphs [8] are being explored in this context. Time expanded graph represents the time-dependence by copying the network for every time instant whereas in time aggregated graphs, the time-varying attributes are aggregated over edges and nodes.
Figure 5(a) shows a network at three time instants. The network topology and parameters change over time. For example, the edge N3-N4 is present at time instants $t = 1, 3$ and disappears at $t = 2$ and its weight changes from 1 at $t = 1$ to 4 at $t = 3$. The time aggregated graph that represents this dynamic network is shown in Fig. 5(b). In this figure, the edge N3-N4 has two attributes, both time series. The attribute [1,3] represents the time instants at which the edge is present and (1,∞,4) is the weight time series, indicating the weights at various instants of time. Figure 5(c) shows the time expanded graph that represents the same scenario. Edge weights in a time expanded graph are not explicitly shown as edge attributes; instead they are



Road Maps, Digital, Figure 5 Two different representations of a time-variant network

represented by edges that connect the copies of the nodes at various time instants. For example, the weight 1 of edge N1-N2 at $t = 1$ is represented by connecting the copy of node N1 at $t = 1$ to the copy of the node N2 at time $t = 2$. The time expansion for the example network needs to go through 7 steps since the latest time instant would end in the network is at $t = 7$. For example, the traversal of edge N3-N4 that starts at $t = 3$ ends at $t = 7$, the travel time of the edge being 4 units.

Cross References

- ► Contraflow in Transportation Network
- ► Emergency Evacuations, Transportation Networks
- ► Nearest Neighbor Queries in Network Databases

Recommended Reading

1. Shekhar, S., Vatsavai, R., Ma, X., Yoo, J.: Navigation Systems: A Spatial Database Perspective. In: Schiller, J. and Voisard, A. (eds.) Location-Based Services, Chapter 3. Morgan Kaufmann, San Francisco, CA, USA, April (2004) ISBN-13: 978-1558609297
2. Elmasri, R., Navathe, S.B.: Fundamentals of Database Systems (5th edn.), Addison Wesley, Boston, MA, USA, March (2006) ISBN-13: 978-0321369574
3. Shekhar, S., Vatsavai, R., Chawla, S., Burke, T.E.: Spatial Pictogram Enhanced Conceptual Data Models and Their Translation to Logical Data Models, Integrated Spatial Databases, Digital Maps, and GIS, Lecture Notes in Computer Science, vol. 1737 (1999)
4. Shekhar, S., Chawla, S.: Spatial Databases: A Tour, Prentice Hall, Upper Saddle River, NJ, USA (2003) ISBN-130: 978-0130174802
5. Hoel, E.G., Heng, W.L., Honeycutt, D.: High Performance Multimodal Networks. In: Proceedings of International Symposium on Spatial and Temporal Databases SSTD (2005)
6. Shekhar, S., Liu, D.R.: CCAM: A Connectivity-Clustered Access Method for Networks and Network Computations. IEEE Trans. Knowl. Data Eng. **9**(1), 102–119 (1997)
7. Kohler, E., Langtau, K., Skutella, M.: Time-Expanded Graphs for Flow-Dependent Transit Times. In: Proceedings of 10th Annual European Symposium on Algorithms (2002)
8. George, B., Shekhar, S.: Time-aggregated Graphs for Modeling Spatio-temporal Networks – An Extended Abstract. In: Proceedings of Workshops at International Conference on Conceptual Modeling(ER), November (2006)
9. Winter, S.: Modeling Costs of Turns in Route Planning. GeoInformatica **6**(4), 363–380 (2002)
10. Anez, J., de la Barra, T., Perez, B.: Dual Graph Representation of Transport Networks. Transp. Res. **30**(3), 209–216 (1996)
11. Caldwell, T.: On Finding minimum Routes in a Network with Turn Penalties. Commun. ACM **4**(2), 107–108 (1961)
12. TomTom, http://www.tomtom.com/ (2007)
13. Garmin, http://www.garmin.com/garmin/cms/site/us (2006)
14. Google Maps, http://maps.google.com (2007)
15. Mapquest, http://www.mapquest.com (2007)

Road Network Data Model

LAURYNAS SPEIČYS, CHRISTIAN S. JENSEN
Department of Computer Science, Aalborg University, Aalborg, Denmark

Synonyms

Transportation network model; Spatial network model; Network data model; Graph; Link-node model; Linear reference model

Definition

A road network data model is a notation that enables the modeling of pertinent aspects of a road-network infrastructure. Using such a notation, a schema of a road-network infrastructure may be designed. This schema may in turn be populated by data, yielding an instance that captures aspects of a specific road network.

Depending of the use context, different aspects of a road-network infrastructure are of interest. A road network data model may consist of several interrelated sub-models, each of which targets the capture of different, specific aspects of a road network. Important examples of such sub-models include the geographical and graph representations and linear referencing.

Geographical representations capture the embedding of a road network into geographical space. Specifically, a road is typically represented by a collection of polylines, where each polyline captures the centerline of part of a road.

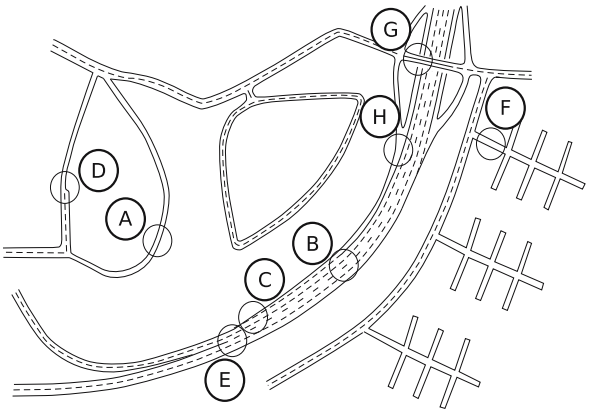
Graph representations, also termed link-node representations, typically aim to capture the topology, or connectivity, of a road network in a compact and computationally efficient format. Graph representations are based on the concepts of undirected and directed mathematical graphs. A node, or vertex, typically models a location with a significant change of traffic properties. Such locations include road intersections. A link, or edge, models the part of the road network that enables travel between two nodes. In a directed-graph model, a directed edge captures that travel between the two nodes involved is allowed in the direction given by the edge. Edge weights capture travel distances or times. A binary so-called co-edge relation models the ability of vehicles to make u-turns in-between intersections. A binary so-called change-edge relation models the ability to make a lane change. To model roads with multiple lanes, multi-graphs that allow multiple edges between a pair of nodes are used.

With linear referencing, a road network is modeled as a collection of one-dimensional linear features that inter-

sect at connections (locations where there is an exchange of traffic). With this model, any location in a road network is given as the identity of a linear feature and a distance from the start of the feature. Such locations may be used for the capture of content, including speed limits and accidents. The kilometer-post representation, an example of a known-marker representation, is often used for the specification of locations: a location is expressed in terms of the identifier of a road, a kilometer post (an example of a distance marker) on the road, and an offset from the distance marker.

With this model, it is also possible to associate a polyline with each linear feature that captures the embedding of the part of the road network modeled by the feature into geographical space.

Sub-models may be integrated by means of procedures that map instances of one sub-model to instances of another sub-model.



Road Network Data Model, Figure 1 Example road network depicting: (A) single-lane, one-way road, (B) multiple lanes in two directions, (C) emergence of a new lane, (D) abrupt change of a bi-directional road into a one-directional road, (E) split of a road into two, (F) restriction-free traffic in a residential area, (G) restricted u-turn, and (H) restricted lane change

Historical Background

Within computer science, the classical approach to the modeling of road networks is to use some variant of either a non-directed or a directed graph.

Both enable the capture of fundamental aspects of a road network, and are mathematically simple and compact formats that are conducive to efficient computation. The prototypical problem addressed in this setting is that of finding shortest paths between locations. In recent work, graphs with different geographical embeddings have been used for the modeling of road networks with the purpose of processing a variety of spatial queries, including variants of range and nearest neighbor queries (e. g., [1,8,11,13]).

So-called navigable data models have been created to support vehicle navigation. Only a few such models capture aspects such as lanes and the connectivities among lanes. Planar [4] and non-planar [3] models have been suggested. The non-planar model, which offers support for data maintenance, captures the geo-location and topology of lanes. The topological information consists of lane connectivities and turn restrictions along lanes and at impedance points. This information can then be used for constructing a graph on which a search can be performed.

In the domain of transportation where, typically, public authorities are concerned with the management (e. g., maintenance) of road-network infrastructures, focus has been on the development of data models that enable the convenient capture and maintenance of road-network related data [2,5,6]. The main objective is to provide means of managing content relevant to administrative tasks.

In this domain, linear referencing [14] has been used quite widely for the capture of content. As an indication of the importance of this domain, Oracle Spatial [10] offers support for linear referencing. In addition, generic schemas, in the form of ER diagrams, have been developed and recommended for the capture of different aspects of entire transportation infrastructures and related content [2,7,9,12,15].

Scientific Fundamentals

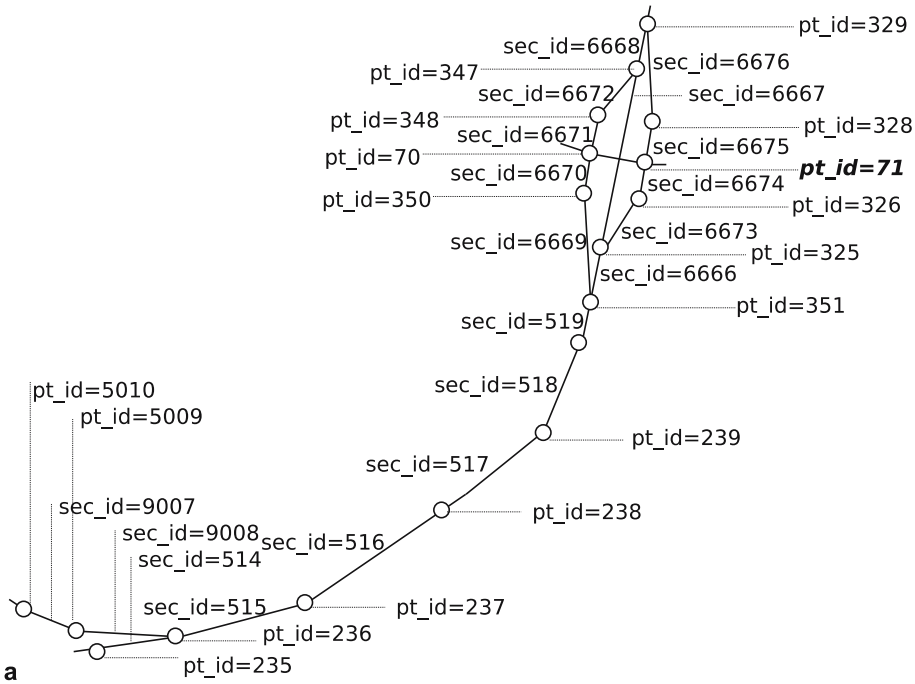
Example Road Network

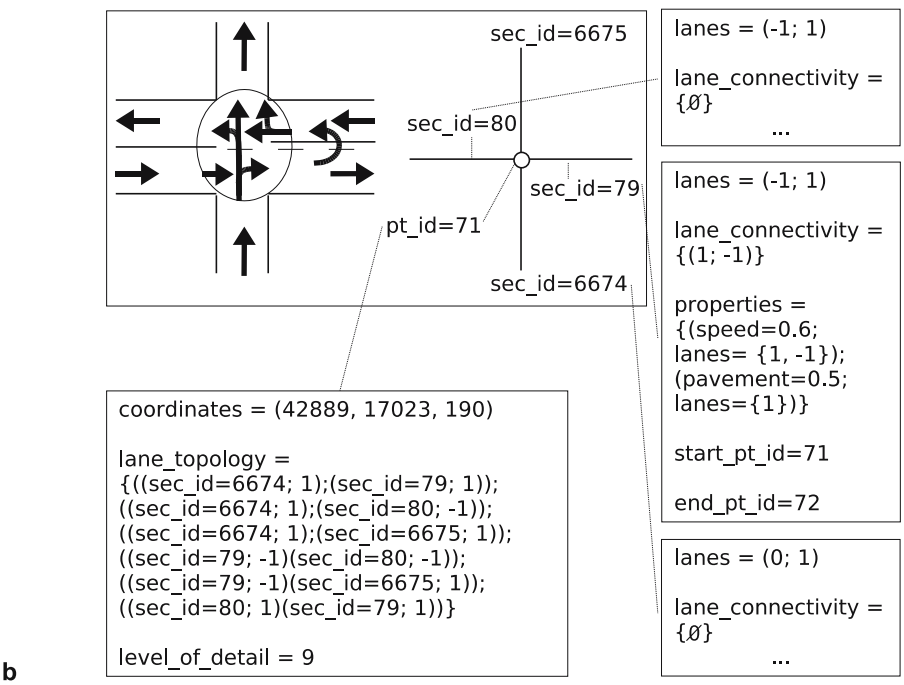
The road network illustrated in Fig. 1 is used for exemplification. The figure depicts a sample road network that embodies a few of the aspects that a road network data model must be capable of capturing. In particular, this network illustrates some of the possible configurations of roads and lanes that are possible in real road networks. Specifically, the network includes bi-directional roads with several lanes in each direction, the appearance and disappearance of a lane due to local access to a highway, and the abrupt disappearance of lanes, i. e., a bi-directional road turning into a single-directional road without reaching an intersection and a road splitting into two.

The figure also illustrates different lane change restrictions. The residential areas to the right have no restrictions on lane changes and u-turns. However, u-turns are not allowed on the bridge that crosses the highway (top right); and on the highway, lane change is prohibited from the access lane for local traffic.

Geographical Road Network Data Models

Geographical data models are used for geo-referencing of the road infrastructure and thereby the road-related content. Specifically, three-dimensional points are the key





Road Network Data Model, Figure 2 Geographical model instance

building blocks of these models. Such points are typically used for the modeling of the center lines of roads. Pairs of such points thus define line segments that model sections of roads. A geographical representation of part of the example road network is shown in Fig. 2a. In the figure, the points are simply associated with identifiers—the coordinates are omitted.

It is often relevant to be able to represent the geography of a road network at different levels of detail. To enable that, an integer value capturing a level of detail may be associ-

ated with each point. The higher the number, the higher the level of detail. Thus, to obtain the representation of a road network at a level of detail corresponding to a certain number, all points with a level of detail number that does not exceed that of the chosen level of detail are considered. As a consequence, the entire sequence of points representing a road network is used when the highest level of detail is chosen.

As mentioned, pairs of consecutive coordinate points model road sections. (Note that consecutive points are allowed to have different levels of detail.) Thus, a road network is partitioned into small sections. In Fig. 2, these are also associated with identifiers. It is possible to associated traffic regulations with each section. For example, it is possible to capture the numbers of lanes in each direction for the section. The allowed movements from one lane to a neighboring lane can also be captured: movement between a pair of lanes may be prohibited/impossible or allowed from one lane to the other only or in both directions.

In an example approach, the lanes in a section are numbered by positive and negative integers, so that the lanes in the one direction are numbered 1, 2, etc. starting with the lane closest to the middle of the road. Similarly, the lanes in the other direction are numbered -1, -2, etc. The number 0 is used to indicate the absence of a lane in one direction. For example, Fig. 2b denotes section 6674 having 1 and 0 lanes in the one and in the other direction, respectively.

For each section, it is also possible to capture properties that affect the movement along the section. Movement-affecting properties are captured if their effect on movement can be quantified. Such properties include speed limits and congestion information. These and a spectrum of other properties can be quantified as conditions that hinder or facilitate movement with respect to some nominal movement condition. The effects of such properties can be normalized over the network. Figure 2b, here, includes a "pavement" property that affects the movement on lane 1 of section 79 by a factor of 0.5 of the nominal movement condition.

Coordinate points either connect two or more sections, or they mark a dead end of a road. For each coordinate point, it is possible to describe how traffic flows between the delimited sections. The possibility to move from a specific lane belonging to one section to a specific lane belonging to the other section is captured by an ordered pair where the first element identifies the former lane and the second element identifies the latter lane. Figure 2b illustrates how turn restrictions between lanes at an intersection are modeled. The figure shows traffic flowing in all directions excluding two u-turns (initiated from section 79, lane -1, and section 80, lane 1) and the left turn from section 80, lane 1 to section 6675, lane 1. Another situation where the description of flow is useful occurs at dedicated spots for u-turn, e. g., on highways with a separating line.

A less obvious use of turn restrictions occurs at points where a coordinate point is placed due to a configuration change on a road. Here lane directions, in most cases, do not change; however, the count of lanes in either of the directions may increase or decrease and the accessibility between lanes may change.

When a coordinate point is present only in order to capture the geographical embedding of a road network at a higher level of detail, the associated flow information is redundant in the sense that removing the coordinate point and its flow information does not result in a loss of flow information.

While the turn restrictions at a coordinate point, e. g., an intersection, can have many configurations, they must satisfy a few requirements. First, for each lane that allows traffic movement into the coordinate point, there must be at least one lane that accepts the incoming traffic; the inverse must be true as well; second, movement coming from a lane that starts at a coordinate point is prohibited, and movement to a lane that ends at a coordinate point is prohibited over the coordinate point.

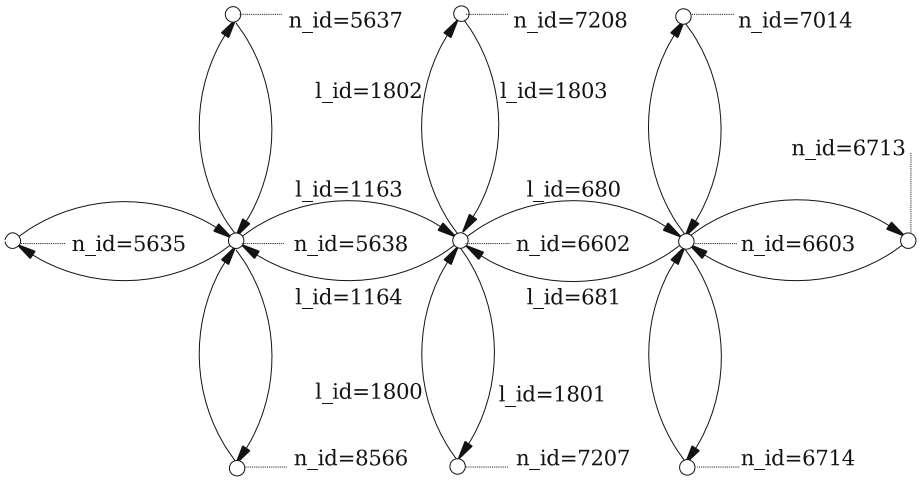
Link-Node Data Models

Link-node, or graph, data models abstract away geographical detail and aim to capture only the topology of a road network. The representations of road networks obtained by using these kinds of models are compact often very well suited as a foundation for query processing.

A typical link-node model captures a road network as a a collection of nodes connected by directed links, i. e., as a directed graph—Fig. 3 illustrates an example instance. (While undirected graphs may be used, they are not considered here.)

A road network may be modeled by several link-node instances that capture the road network at different levels of detail. For example, a certain region may be modeled by two link-node instances: a very detailed one used for detailed route planning and a less detailed one used for, e. g., higher-level route planning.

Thus, a node may be used in one or several instances. Conceptually, the use of a node is identified by a pair of attributes: a unique network identifier and a unique node identifier within the scope of the network. Similarly, a link may be used in several instances. Thus, usages of links have attributes analogous to those of nodes, i. e., the pair of a network identifier and a link identifier within the instance. Moreover, each link has a start node and an end node that belong to the same instance. Further, each



Road Network Data Model, Figure 3 A link-node model instance

link has a length that is normally constrained to be non-negative.

The link-node instance in Fig. 3 captures part of the road network depicted in Fig. 4 in a manner appropriate for high-level route planning. The links represent the routes, not individual roads, e. g., links 1163 and 1164 represent the forward and backward routes between the first and second intersections, respectively. For this reason, the complex intersections, i. e., the two over-passes and the rotary are each reduced to a single node, i. e., to the nodes 5638, 6602, and 6603.

In the example, the length for each link is approximately equal to the length of the corresponding route (in meters). In general, the length values may be given more complex semantics, e. g., the minimum travel time that is needed to traverse the corresponding route. A directed link in the example indicates that one node can be reached from another node. Thus, a route that corresponds to a bi-directional road is represented by a pair of oppositely directed links.

To capture a road network in a format that is appropriate for low-level navigation, a simple directed graph is not adequate. Instead, a directed multi-graph, i. e., a directed graph that can have more than one link (in the same direction) between a pair of nodes, is used.

A route followed by a vehicle through a road network may be captured using a sequence of full edges. However, it is at times allowed for a vehicle to make a u-turn, or to change lanes in-between nodes. To capture such behavior more accurately, one can introduce so-called co-edge and change-edge functions that model u-turns and changes of lanes.

A co-edge function captures pairs of edges that represent pairs of lanes for which it is allowed to make a u-turn from the first lane of the pair to the second lane. Intuitively, a moving object may jump from an edge to its co-edge without having reached a vertex. A u-turn can be made only to the opposing traffic lane that is closest to the current traffic direction. A moving object may overtake a car by temporarily entering an oppositely directed lane if that lane is a co-edge of the object's current lane.

Similarly, the change-edge function captures pairs of edges for which it is possible to change from the argument lane to the result lane. Lane changes are restricted to lanes in the same traffic direction. The intuition is the same as for u-turns—that of a moving object being able to jump from one lane to another without visiting a vertex.

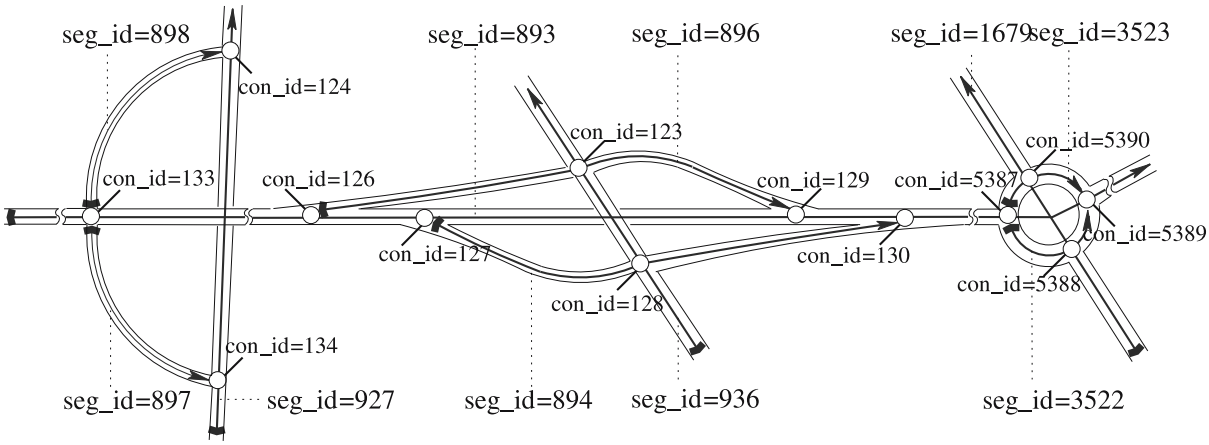
Linear Referencing Data Model

A linear referencing data model of a road network is a collection of one-dimensional so-called linear features, that intersect at connections. This representation is illustrated graphically in Fig. 4.

Each linear feature has a unique identifier and a length, and any position on the feature can be expressed as a distance from the start of the feature. The length of a linear feature is equal to the length of the corresponding part of the road network. The embedding of a linear feature into space can be, and typically is, described by a polyline, in a manner similar to what was described for the geographical data models.

Connections model intersections and also have unique identifiers. Several linear features are involved in a connection. For each such feature, the connection occurs at some distance from the start of the feature. It should be noted that the positioning of content, e. g., connections and accident reports, along a feature is independent of the polyline chosen for capturing the geographical location of the feature.

Figure 4 depicts an example of linear referencing-based model of part of a real road network [5]. In the example,



Road Network Data Model, Figure 4 Linear feature road network instance

the road network is modeled at a high level of detail. Each linear feature generally is as long as possible while preserving the network topology. For example, segment 893 corresponds to the "long" main road and is 78,326 meters long. In order to preserve the topology of the network, segment 3522, which is only 62 meters long, is assigned to the bottom semi-circumference of the rotary to the right, which connects two disjoint sections of the main road. As can be seen, connections are placed at road intersections.
When modeling a road network using linear referencing, the linear features should partition the road network. Long features are preferable because they lead to a more compact segment representation and, more importantly, a more compact and thus update-friendly representation of the associated content.

Key Applications

Typical applications of road network data models relate to different aspects of transportation.

Road Planning

Link-node data models are used for road planning, which refers to the (re)designing of road networks while taking, e. g., traffic flows into consideration.

Road Management

Kilometer-post based models (the most commonly used type of known-marker based models) are used for road administration. This type of model is useful for collecting and utilizing data in the field.
With this type of model, a location expressed in terms of a road, a distance marker on the road (i. e., a kilometer post), and an offset from the distance marker can be used for uniquely positioning of content. Primitive technological means, such as a simple measuring device and a map and a ruler, then suffice for identifying a position on a road.

Route Planning Services

Link-node type models are used for tasks such as route planning. The task refers to the retrieval of traversable routes that satisfy certain criteria from a road network. Directed graphs that capture traffic regulations are appropriate for this task.

In-Vehicle Navigation

One class of in-vehicle services concerns navigation. These services determine the most suitable route to a destination and provide instructions for real time navigation. For high-quality services, instructions take into account traffic regulations at the granularity of lanes. Another class of in-vehicle services relate to safety. Here, services warn drivers about possible collisions and assist the drivers in various ways. In order to provide this functionality, the services may rely on models that capture the underlying road network at lane granularity.

Mobile E-Services

Internet-worked mobile devices such as mobile phones, personal digital assistants, and navigation devices enable a range of new personal information services, many of which will exploit information about the user's geolocation for providing the desired functionality.
Example services include finder applications that allow the users to locate friends or family, businesses, or landmarks. Services may also deliver maps, directions, or traffic reports. Services may involve the identification of a service user's nearest neighbors of a certain kind. An example service may identify the emergency room that is within the

closest driving distance; another may reserve the taxi that is nearest to the service user.

Such services concern objects moving in and located in road networks, as well as content reachable via road networks. Thus, road network models are needed for the underlying computations.

Future Directions

The detailed modeling of road networks is becoming increasingly relevant. Specifically, advances in positioning technologies are slated to enable the positioning of vehicles within lanes.

For example, the Galileo positioning system will offer better positioning than does GPS with respect to several aspects, including the accuracy, penetration, and time to fix. For example, the best-case accuracy (without the use of ground stations) of Galileo is 45 cm as opposed to 2 m for GPS. Next generation GPS will also offer better positioning, and Galileo and GPS are expected to be interoperable. As another example, infrastructures that rely on in-road and in-vehicle sensors for accurate positioning at lane resolution are being conceived in the telematics community.

Exploitation of the capability of positioning at lane resolution will enable increases in the qualities of existing services, e. g., navigation services, but will also enable entirely new services, e. g., collision warning and assisted driving services.

Road network data models that capture road-network infrastructures at the lane level are essential in many applications that may exploit lane-level positioning. Such models serves as a basis for data retrieval, i. e., for query processing, which is the topic of much recent and on-going work. For example, with the objective of supporting different location-based services, much attention is being dedicated to the efficient support for a variety of proximity queries—including conventional, skyline, and trajectory-based queries, for static as well as moving objects. This line of research may lead to both future modifications of the model, to new insights into query processing, and to new and exciting applications.

Cross References

- ▶ ArcGIS: General Purpose GIS Software System
- ▶ Contraflow in Transportation Network
- ▶ Data Compression for Network GIS
- ▶ Emergency Evacuations, Transportation Networks
- ▶ Location-Based Services: Practices and Products
- ▶ Nearest Neighbor Queries in Network Databases
- ▶ Network GIS Performance
- ▶ Oracle Spatial, Geometries
- ▶ Road Maps, Digital
- ▶ Routing Vehicles, Algorithms
- ▶ Routing Vehicles, Algorithms
- ▶ Trip Planning Queries in Road Network Databases
- ▶ Voronoi Diagrams for Query Processing

Recommended Reading

1. De Almeida, V.T., Güting, R.H.: Using Dijkstra's Algorithm to Incrementally Find the k-Nearest Neighbors in Spatial Network Databases, In: Proc. ACM Symposyum on Apagesied Computing, pp. 58–62 (2006)
2. Dueker, K., Butler, J.A.: GIS-T Enterprise Data Model with Suggested Implementation Choices, J. Urban Reg. Inform. Syst. Assoc. **10**(1), 12–36 (1998)
3. Fohl, P., Curtin, K.M., Goodchild, M.F., Church, R.L.: A Non-Planar, Lane-Based Navigable Data Model for ITS, In: Proc. International Symposium on Spatial Data Handling, pp. 17–29 (1996)
4. Gottsegen, J., Goodchild, M., Church, R.: A Conceptual Navigable Database Model for Intelligent Vehicle Highway Systems, In: Proc. GIS/LIS, pp. 371–380 (1994)
5. Hage, C., Jensen, C.S., Pedersen, T.B., Speicys, L., Timko, I.: Integrated Data Management for Mobile Services in the Real World, In: Proc. International Conference on Very Large Data Bases, pp. 1019–1030 (2003)
6. Jensen C.S.: Database Aspects of Location-Based Services, In: Schiller, J., Voisard, A.: Location-Based Services, Morgan Kaufmann Publishers, pp. 115–148 (2004)
7. Koncz, N., Adams, T.M.: A Data Model for Multi-Dimensional Transportation Location Referencing Systems, J. Urban Reg. Inform. Syst. Assoc. **14**(2), 27–41 (2002)
8. Kolahdouzan, M., Shahabi, C.: Voronoi-Based K Nearest Neighbor Search for Spatial Network Databases, In: Proc. International Conference on Very Large Data Bases, pp. 840–851 (2004)
9. NCHRP: A Generic Data Model for Linear Referencing Systems, Transportation Research Board, Washington, DC (1997)
10. Murray, C.: Oracle Spatial User Guide and Reference, Release 9.2., Oracle Corporation (2002)
11. Papadias, D., Zhang, J., Mamoulis, N., Tao, Y.: Query Processing in Spatial Network Databases, In: Proc. International Conference on Very Large Data Bases, pp. 802–813 (2003)
12. Okunieff, P., Siegel, D., Miao, Q.: Location Referencing Methods for Intelligent Transportation Systems (ITS) User Services: Recommended Apagesoach, In: Proc. GIS-T Conference, pp. 57–75 (1995)
13. Sankaranarayanan, J., Alborzi, H., Samet, H.: Efficient Query Processing on Spatial Networks, In: Proc. ACM International Workshop on Geographical Information Systems, pp. 200–209 (2005)
14. Scarponcini, P.: Generalized Model for Linear Referencing in Transportation, GeoInformatica **6**(1), pp. 35–55 (2002)
15. Zeiler, M.: Modeling Our World, ESRI Press (1999)

Road Networks

- ▶ Contraflow in Transportation Network
- ▶ Spatio-temporal Queries on Road Networks, Coding Based Methods

Roadway Network Model

► Emergency Evacuations, Transportation Networks

Root-Mean-Square Error

► Photogrammetric Products

Rough Approximation

► Approximation

Rough Set Theory

► Approximation

Route Activity

► Crime Mapping and Analysis

Routing

► Data Collection, Reliable Real-Time

Routing Vehicles, Algorithms

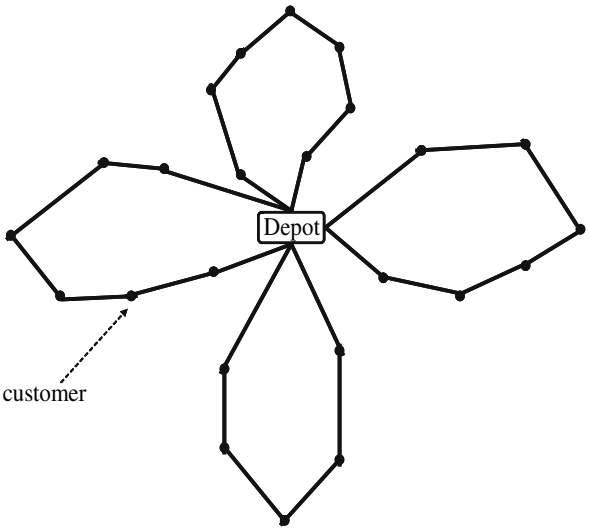
CHRISTOS D. TARANTILIS
Department of Management Science & Technology,
Athens University of Economics & Business,
Athens, Greece

Synonyms

Distribution logistics; Fleet management; Vehicle routing problem; Simulated annealing; Threshold accepting method; Record-to-record travel method

Definition

The Vehicle Routing Problem (VRP) [1] embraces a class of complex combinatorial optimization problems that target the derivation of minimum total cost routes for a number of resources (vehicles) located at a central point (depot) in order to service efficiently a number of demand points (customers). The standard version of VRP (known as basic VRP) is defined on a graph $G=(V,A)$, where $V=\{u_0, u_1, \ldots, u_n\}$ is the vertex set and $A=\{(u_i, u_j): u_i, u_j \in V, i \neq j\}$ is the arc set of G. Vertex u_0 represents a depot (warehouse or distribution centre) that hosts a homogeneous fleet of m vehicles with capacity Q. The remaining vertices correspond to demand points (or equivalently, customers). Each customer u_i has a non-negative demand q_i. The vector of all customer demands is denoted by $q(V)$. Furthermore, a non-negative cost matrix $C=(c_{ij})$ is defined on A; usually, the cost c_{ij} models the travel time between customers u_i and u_j. If $c_{ij} = c_{ji}$, the problem is symmetric, and it is common to replace A with the edge set $E=\{(u_i, u_j): u_i, u_j \in V, i \neq j\}$. The solution to the basic VRP is a set of routes that satisfy the following constraints: a) each route starts and ends at the central depot; b) each customer is visited exactly once; c) every customer's demand is satisfied; d) the total travel time of the set of routes is minimized.



Routing Vehicles, Algorithms, Figure 1 A typical VRP solution

The aim of this chapter is to focus on the annealing-based solution approaches for solving two of the most studied VRP types: the Capacitated VRP (CVRP) and the Distance Constrained VRP (DCVRP) [2]. The additional constraints imposed to model the routing scenarios of the aforementioned VRP types are:

- the total demand of the customers covered by a route cannot exceed the capacity of a vehicle Q (for both CVRP and DCVRP);
- the total travel time of any vehicle route cannot exceed a pre set upper bound (only for DCVRP).

The objective (for both CVRP and DCVRP) is to minimize the sum of travel time. It is important to note that the number of vehicles is either pre-determined or is treated as a decision variable.

Historical Background

In the late 70s, several algorithms were developed for solving VRPs for very small numbers of variables and constraints. Later, since exact algorithms were not capable of consistently solving instances with more than 50 customers, heuristics were mainly employed for real-life medium and large-scale vehicle routing problems [3].

Heuristics can be divided into two classes: *classical heuristics* that perform relatively limited exploration of the search space to produce good solutions fast, and *metaheuristics* that are general-purpose mechanisms guiding intelligently the search process, combining neighborhood search rules, memory structures and recombination of solutions [4].

Classical heuristics for both the CVRP and DCVRP can be broadly classified into three categories [3]:

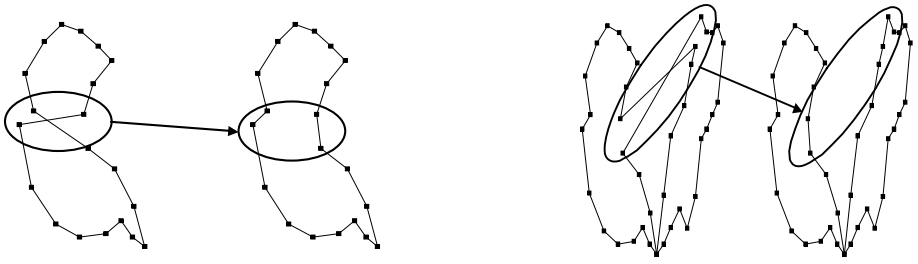
a) *Construction heuristics* build (guided by some cost minimization criterion) a solution, selecting a solution component (vertex or arc) step by step until the partial solution is completed. The solution can be constructed sequentially (i. e. producing one route at a time) or parallel (i. e. producing several routes simultaneously). The myopic approach of adding the best solution component according to the least cost increase is called *greedy* approach.

b) *Two-phase heuristics* produce a feasible solution in the first phase and then optimize the sequence of customers on each route in the second phase.

c) *Local search classical heuristics* used to improve a VRP-solution by introducing changes in the current solution. Local search is an iterative search procedure that, starting from an initial feasible solution, progressively improves it by applying a series of local modifications called *moves*. At each iteration of local search, the set of *moves* that can be applied to the current solution s, define a set of neighboring solutions denoted $N(s)$. More specifically, $N(s)$ is a subset of search space (i. e. space of all possible solutions than can be visited during the search) which consists of solutions generated by applying a single transformation to the current solution s. According to local search rationale, at each iteration, the search moves to an improving neighbour-feasible solution until it will get trapped in a local optimum, which usually represents a low quality solution.

Three of the most applicable moves are the *2-Opt*, 1–1, and *1–0 Exchange* moves [3].

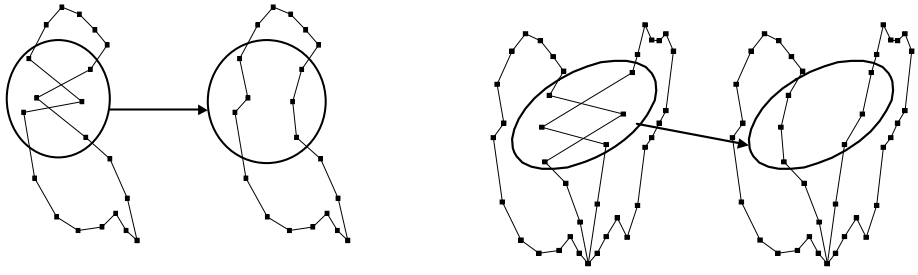
A 2-Opt *move* operates as follows: Suppose a route $R = \{u_0, u_1, \ldots, u_n, u_0\}$ of a solution s, and let $Z = \{(u_i, u_{i+1}); (u_j, u_{j+1})\}$ be a set of two edges in R that form criss-cross. A 2-Opt *move* eliminates then the criss-cross and reverses a section of the R by replacing the edges of Z by edges of $W = \{(u_i, u_j); (u_{i+1}, u_{j+1})\}$ to reconstruct the route R. In the multiple routes, edges (u_i, u_{i+1}) and (u_j, u_{j+1}) belong to different routes but they form a criss-cross again. A 2-Opt *move* is applied exactly in the same way as in the case of single route. This is demonstrated in Fig. 2 [2].



2-Opt move for single route

2-Opt move for multiple routes

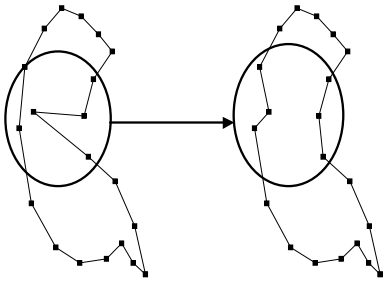
Routing Vehicles, Algorithms, Figure 2 The 2-Opt move



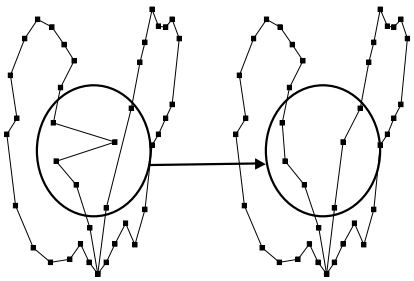
1-1 Exchange move for single route

1-1 Exchange move for multiple routes

Routing Vehicles, Algorithms, Figure 3 The 1–1 Exchange move



1-0 Exchange move for single route



1-0 Exchange move for multiple routes

Routing Vehicles, Algorithms, Figure 4 The 1–0 Exchange move

In 1–1 Exchange *move*, two vertices from either the same or different routes are swapped, while in 1–0 Exchange *move*, a vertex is transferred from its position in one route to another position in either the same or a different route. These moves are shown in Figs. 3 and 4 respectively [2].

As a major departure for the classical heuristics, metaheuristics are non problem specific approximate algorithms either stochastic or deterministic which combine, guide and subordinate classical heuristic methods in higher level frameworks, using different types of memory structures and learning mechanisms, as well as analogies with optimization methods found in nature. Metaheuristics typically produce much higher quality solutions than those obtained with classical heuristic approaches [5].

Annealing-based algorithms belong to the first metaheuristics extending classical local search by employing explicit strategies to escape from local optima, allowing moves to inferior solutions. The term "annealing" is due to the fact that conceptually the algorithms of this category inspired by a physical process, known as annealing, where a material is heated into a liquid state then cooled back into a recrystallized solid state [6]. The main advantages of annealing-based algorithms are their simple structure, general applicability and computational effectiveness on different combinatorial optimization problems.

In the following section of this chapter, the implementations of the most effective annealing-based metaheuristics to the solution of the CVRP and DCVRP are reported.

Scientific Fundamentals

A description of the basic principles of the annealing-based metaheuristic approaches is given first, followed by a description of the implementations to the CVRP and DCVRP.

Simulated Annealing – Basic Principles

Simulated Annealing (SA) [6] finds its inspiration by physical annealing process studied in statistical mechanics. SA conducts local search while offering the possibility of accepting, in a controlled manner, worse solutions. This feature allows SA to escape from a low quality local optimum.

More precisely, at each iteration t of SA, a neighbour $s' \in N(s)$ of the current solution s is generated stochastically and a decision is then made to decide whether s' will replace s. If s' is better than s i. e. $\Delta = c(s') - c(s) \leq 0$ (for a minimization problem), the search moves from s to s', otherwise, the search is moved to s' with the probability $e^{(-\Delta)/\theta t}$. This probability depends on a) the degree of the degradation, and b) a control parameter θ called temperature (higher temperatures lead to higher accepting probabilities and vice versa). The temperature is controlled by a cooling schedule specifying how the temperature should be progressively reduced. Typically, SA stops when a fixed number of non-improving iterations is realized with the temperature or when a pre-specified number of iterations is reached. The SA algorithm is summarized in the Fig. 5.

R

Simulated Annealing Algorithms for the CVRP and DCVRP

Osman [7] implemented a SA algorithm for solving the CVRP and DCVRP, using a systematic way to explore the neighbourhoods of the current solution. Route pairings were first constructed in the following order: $(R_{\pi(1)}, R_{\pi(2)}), \ldots, (R_{\pi(1)}, R_{\pi(m)}), (R_{\pi(2)}, R_{\pi(3)}), \ldots, (R_{\pi(m-1)}, R_{\pi(m)})$ where π was a permutation of customers and m was the number of vehicles used. Then, a λ-*interchange* neighbourhood structure was employed, in which exchanges of customers for each pairing took place until an improvement was identified. A sophisticated cooling schedule was also applied: the temperature θ_t at iteration t decreased continuously as long as the current solution was modified, while it was either halved or replaced by the temperature at which the incumbent was identified whenever the local search in the inner loop was completed without accepting any λ-*interchange* move.

Simulated Annealing

Initialization: Step 1. Produce an initial feasible solution s

Step 2. Select an initial temperature $\theta_1 > 0$.

Outer Loop: While Outer Loop criterion NOT satisfied do

Inner Loop: While Inner Loop criterion NOT satisfied do

i) Generate solution $s' \in N(s)$);

ii) If $\Delta = c(s') - c(s) \leq 0$, then // *Acceptance*

{set $s = s'$; check if $c(s) < c(s_{best})$ then $s_{best} = s$;}

iii) If $\Delta = c(s') - c(s) > 0$, then $s = s'$ with probability $\exp(-\Delta / \theta_t)$.

Repeat Inner Loop:

If s' replaces s at least once in inner loop,

then decrease temperature θ_t { $\theta_t = \theta_{t+1}$; $\theta_{t+1} = \alpha_1 * \theta_t$; }

Repeat Outer Loop:

Report best solution found;

Routing Vehicles, Algorithms, Figure 5 The Simulated Annealing algorithm

Threshold Accepting

Initialization: Step 1. Produce an initial feasible solution s

Step 2. Select an initial threshold $T_{ho} > 0$.

Outer Loop: While Outer Loop criterion NOT satisfied do

Inner Loop: While Inner Loop criterion NOT satisfied do

Generate solution $s' \in N(s)$;

If $c(s') - c(s) \leq T_h$, then // *Acceptance*

{set $s = s'$; check if $c(s) < c(s_{best})$ then $s_{best} = s$;}

Repeat Inner Loop.

If s' replaces s, i.e. if $c(s') - c(s) \leq T_h$ is satisfied at least once in inner loop, then reduce threshold

{ $T_{old} = T_{new}$; $T_{new} = \alpha_1 * T_{old}$; } // Threshold reduction

Repeat Outer Loop.

Report best solution found;

Routing Vehicles, Algorithms, Figure 6 The Threshold Accepting algorithm

Threshold Accepting – Basic Principles

Threshold accepting (TA) [8] is a modification of the SA. More precisely, it leaves out the stochastic element in accepting worse solutions by introducing a deterministic threshold, denoted $T_h > 0$, and accept a worse solution if $c(s') - c(s) \leq T_h$ (the inequality represents the *move acceptance criterion*). During the optimization process the threshold level is gradually lowered like the temperature in SA. As long as the value of T_h is high, the local search performed is not goal oriented, thus achieving high diversification (i. e. elaborating different regions in the solution space) and low intensification (i. e. concentrating the search into a specific region of the search space) of the search process. However, as the search procedure evolves and T_h is reduced, the balance between diversification and intensification changes until the typical threshold accepting algorithm behaves nearly like a descending local search algorithm (i. e. accepts only cost-improving solutions). The TA algorithm is summarized in the Fig. 6.

Threshold Accepting Algorithms for CVRP and DCVRP

Tarantilis et al. [9,10] developed two TA-based algorithms for solving CVRP and DCVRP called Backtracking Adaptive Threshold Accepting (BATA) and List Based Threshold Accepting (LBTA) respectively. The basic innovation of BATA over the standard TA method was that T_h was not necessarily reduced in a monotonic fashion during the search process but also incorporated an occasional increase, called backtracking, of its value. The adopting of

BATA

Initialization: Step 1. Produce an initial feasible solution s
Step 2. Select an initial threshold $T_{ho} > 0$.

Outer Loop: While Outer Loop criterion NOT satisfied do

Inner Loop: While Inner Loop criterion NOT satisfied do

Generate solution $s' \in N(s)$;

If $c(s') - c(s) \leq T_h$, then // *Acceptance*

{set $s = s'$; check if $c(s) < c(s_{best})$ then $s_{best} = s$;}

Repeat Inner Loop.

If s' replaces s, i.e. if $c(s') - c(s) \leq T_h$ is satisfied at least once in inner loop, then reduce threshold

{ $T_{old} = T_{new}$; $T_{new} = \alpha_1 * T_{old}$; } // Threshold reduction

else

raise the threshold value (backtracking)

Repeat Outer Loop.

Report best solution found;

LBTA

Initialization: Step 1. Produce an initial feasible solution
Step 2. Compute the threshold values that represent the initial List by conducting local search

Loop: While Loop criterion NOT satisfied do

New acceptances based the maximum threshold value, $T_{\max}$, stored in the List at every iteration

(a) Generate s' from s. Compute $T_{new} = \dfrac{c(s') - c(s)}{c(s)}$

(b) If $T_{new} = \dfrac{c(s') - c(s)}{c(s)} \leq T_{\max}$ then {set $s = s'$; check if $c(s) < c(s_{best})$ then $s_{best} = s$;}. If (s has changed then insert T_{new} in List) {Insert: $T_{new} \rightarrow$ *List;* Pop: *List* $\rightarrow T_{max}$}

(c) Repeat **Loop**;

Report best solution found;

Routing Vehicles, Algorithms, Figure 7 The BATA and the LBTA algorithms

Record to Record Travel

Initialization: Step 1. Produce an initial feasible solution s
Step 2. Select a Deviation $D > c(s)$. Let $c(s_{best}) := c(s)$

Loop: While stop criterion NOT satisfied do

i) Generate solution $s' \in N(s)$;

ii) If $c(s') < c_{best} + D$ then // *Acceptance*

{set $s = s'$; check if $c(s) < c(s_{best})$ then $s_{best} = s$;}

Repeat Loop.

Report best solution found;

Routing Vehicles, Algorithms, Figure 8 The Record-to-Record Travel algorithm

this non-monotonic schedule, results in an oscillating strategy that achieved a dynamic balance between diversification and intensification of the search process. Regarding LBTA, its basic innovation over the standard TA method was the introduction of a List of threshold values, which were used in the *move acceptance criterion* expressed as $\frac{c(s')-c(s)}{c(s)} \leq T_{\max}$ where $T_{\max}$ was the maximum threshold value stored in the list, and helped the method decide whether s' will replace s.

More precisely, the List served as a memory of the variability of local function values, stored in the form of value changes from each old configuration. The introduction

of the List of threshold values also helped the designer of a VRP-algorithm reduce the parameters involved in the threshold reduction strategy, since parameters (such as the initial value of threshold and the percentage of the threshold reduction) were determined automatically.

Record-to-Record Travel – Basic Principles

The Record-to-Record Travel algorithm [11] accepts the neighbor s' of the current solution if it is not much worse than the best solution found so far during the optimization process. The cost of the best solution is called *Record*. The *Deviation* is the only parameter of the algorithm, defined as $k\% \times Record$. The Record-to-Record Travel algorithm is summarized in the Fig. 8.

Record-to-Record Travel Algorithms for CVRP and DCVRP

Golden et al. [12] developed a Record-to-Record Travel (RtRT) algorithm for solving large-scale vehicle routing problems. The proposed methodology constructed an initial solution by applying the Clarke and Wright savings heuristic [3]. The 1–0 and 1–1 exchange feasible moves were then employed within the record-to-record travel procedure. To clean up routes produced, only cost-improving moves were then used. The individual routes were then re-sequenced and the operations of 1–0 exchange, 1–1 exchange and clean-up were repeated. Whenever the solution has not improved for a number of iterations, the best solution was perturbed by reinserting some of its vertices in different positions and repeating the operations of 1–0, 1–1 exchanges and clean-up. The Golden et al. RtRT algorithm is sketched in Fig. 9.

The RtRT metaheuristic developed by Li et al. (2004) [13] was a modification of the Golden et al. [12] algorithm. The algorithm was called VRtRT due to the variable-length neighbor list used within the RtRT procedure. This idea, which was inspired by the Granular Tabu Search algorithm [4], was to (a priori) delete from the examined graph long edges that were unlikely to be part of the optimal solution. Following this rationale, the VRtRT examined only a fixed number of neighbors for each vertex when employing the operations of 1–0 exchange, 1–1 exchange and 2-opt moves. These neighbors determined by a proportion p of the 40 shortest edges incident to each vertex. The parameter p varied during the optimization process.

Computational Results

The metaheuristic algorithms described in this Section, as far as we know, constitute the most effective annealing-based algorithms (in terms of solution quality) for solving the CVRP and DCVRP, according to their performance on the following two well-known sets of benchmark instances:

- The fourteen benchmark instances generated by Christofides et al. [14]. Each instance contains between 50 and 199 nodes as well as the depot. The location of the nodes is defined by their Cartesian co-ordinates, and the travel cost from vertex u_i to u_j is assumed to be the respective Euclidean distance. Problems 1–5, 11 and 12 are CVRPs while the problems 6–10, 13 and 14 are DCVRPs. For the first ten problems, vertices are randomly located over a square, while for the remaining ones, vertices are distributed in clusters and the depot is not centered;
- The twenty large scale vehicle routing problems (LSVRPs) proposed by Golden et al. [12]. These instances contain between 200 and 483 customers while the first eight of them have route-length restrictions (i. e. DCVRPs).

The problem instances can be found at http://neo.lcc.uma.es/radi-aeb/WebVRP/.

Close examination of the results demonstrated in Tables 1– 4 shows that the concept of VRtRT introduced by Li et al. [13] is probably the most powerful idea put forward in the area of annealing-based metaheuristics for solving CVRPs and DCVRPs in recent years. However, LBTA [10] manages to produce competitive results with VRtRT, using just one parameter within its structure.

It is also noteworthy that the variable-length neighbor list helps VRtRT both speed up the search process and produce

The RtRT algorithm of Golden *et al* [12]
Step 1. Produce an initial solution by Clarke and Wright
Step 2. Apply feasible 1-0 exchange moves and record-to-record travel.
Step 3. Apply feasible 1-1 exchange moves on different routes and record-to-record travel.
Step 4. Apply feasible 1-0 exchange, 1-1 exchange and 2-opt moves (only cost-improving moves)
Step 5. Apply local reinitialization: Repeat R_1 times and resequence individual routes. Go to Step 2.
Step 6. Repeat R_2 times and pertub the current best solution. Go to Step 2.

Routing Vehicles, Algorithms, Figure 9 The RtRT algorithm of Golden et al. [12]

Routing Vehicles, Algorithms, Table 1 Computational performance of the best known annealing-based metaheuristics on the benchmark instances generated by Christofides et al. [14]

Pr.	Osman	BATA	LBTA	RtRT	VRtRT
	Value	Value	Value	Value	Value
1	528	524.61	524.61	–	524.61
2	838.62	839.56	838.18	–	836.18
3	829.18	830.34	830.21	–	827.39
4	1058	1037.17	1036.05	–	1045.36
5	1378	1318.49	1317.81	–	1303.47
6	555.43	555.43	555.43	–	–
7	909.68	909.68	909.68	–	–
8	866.75	868.58	867.41	–	–
9	1164.12	1174.60	1173.89	–	–
10	1417.85	1418.27	1421.01	–	–
11	1176	1042.11	1042.11	–	1042.11
12	826	819.56	819.56	–	819.56
13	1545.98	1547.74	1547.28	–	–
14	890	866.37	866.37	–	–

Routing Vehicles, Algorithms, Table 2 Average computing comparison (in minutes) of four annealing-based meta- heuristics on the benchmark instances generated by Christofides et al. [14]

	Osman SAVAX 8600	BATAPentium II 400 MHz	LBTAPentium II 400 MHz	VRtRT Athlon 1 GHz
	Time	Time	Time	Time
Average Computing time	151.4	6.5	6.8	0.41

Routing Vehicles, Algorithms, Table 3 Performance of four annealing-based meta-heuristics on large scale problem instances generated by Golden et al. [12]

Pr.	Osman	BATA	LBTA	RtRT	VRtRT
	Value	Value	Value	Value	Value
1	–	5683.63	5680.16	5834.60	5666.42
2	–	8528.80	8512.64	9002.26	8469.32
3	–	11199.72	11190.38	11879.95	11145.80
4	–	13661.16	13706.78	14639.32	13758.08
5	–	6466.68	6460.98	6702.73	6478.09
6	–	8429.28	8427.72	9016.93	8539.61
7	–	10297.27	10274.19	11213.31	10289.72
8	–	11953.93	11968.93	12514.20	11920.52
9	–	596.92	595.35	587.09	588.25
10	–	765.03	764.88	749.15	749.49
11	–	945.20	945.09	934.33	925.91
12	–	1143.39	1143.74	1137.18	1128.03
13	–	872.66	871.97	881.04	865.20
14	–	1102.40	1102.66	1103.69	1097.78
15	–	1384.04	1385.59	1364.23	1361.41
16	–	1679.50	1677.25	1657.93	1635.58
17	–	718.16	717.40	720.44	711.74
18	–	1030.54	1032.07	1029.21	1010.32
19	–	1408.62	1406.47	1403.05	1382.59
20	–	1872.23	1872.87	1875.17	1850.92

Routing Vehicles, Algorithms, Table 4 Average computing comparison (in minutes) of four annealing-based meta- heuristics on large scale problem instances generated by Golden et al. [12]

	BATAPentium II 400 MHz	LBTAPentium II 400 MHz	RtRTPentium 100 MHz	VRtRTAthlon 1 GHz
	Time	Time	Time	Time
Average Computing time	18.4	17.8	37.2	1.13

much better results than the RtR algorithm (due to focus on promising moves).

Key Applications

Regarding the practical interest, in today's competitive business environment, VRP is not just a model of moving goods, but rather a critical model in effectively managing and operating the supply chain. In particular, VRP is involved in:

Transportation

The VRP model is one of the models that holds the supply chain together and companies that invest in research for developing effective algorithms will definitely improve their transportation management practices and will undoubtedly generate strategic benefits for themselves and their supply chain partners.

E-commerce

"People don't buy products, they buy delivered products," highlight supply chain experts. That's why it is essential to tie e-commerce to a distribution network. It is understandable that the growth of e-commerce and its distribution needs are inevitable. Customers are buying their products online because of convenience and speed of delivery. If e-commerce companies can't deliver products in a timely matter, customers either will buy from an online competitor or from the local retail store. In addition, as online sales continue to increase, more and more companies will be faced with the dilemma of partnering with a logistics company that will provide a cost-effective and seamless distribution solution. The companies that don't step up to the plate may find themselves as obsolete as the e-commerce companies that fail to create affordable and efficient distribution networks.

Reverse Logistics

In addition to the distribution process to the customers, re-usable packaging and goods to be recycled or remanufactured have to be transported in the reverse direction. On a strategic level, design decisions for the reverse logistics system have to be taken: Who performs which task and where. On a medium-term level the operator of the redistribution system and the relation between forward and reverse channel (thus distribution and redistribution system) have to be determined. If the decision is made that the foreword and reverse are to be run independently, for each of the channels a separate VRP has to be solved.

Green and Risk Logistics

The VRP methodology plays an important role in Green and Risk Logistics operations by finding routes for the transportation of hazardous materials (i. e. transportation of gas cylinders) [15] such that the population exposure risk or/and the of the environment pollution risk is mitigated.

Future Directions

The high quality solution produced by VRtRT, LBTA and BATA encourages for future work on designing annealing-based metaheuristics with simple structure, few parameters and intelligent strategies for solving real-time vehicle routing problems (RT-VRPs). These constitute a generic class of dynamic and mix of dynamic and stochastic VRPs aiming at quick reaction within a prescribed time schedule, in response to the incomplete or uncertain information revealed (customer requests, travel times, breakdowns, etc) as routes are executed.

Recommended Reading

1. Toth, P., Vigo, D.: The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia (2002)
2. Tarantilis, C.D., Kiranoudis, C.T.: Using a spatial decision support system for solving the vehicle routing problem. Inf. Manag. **39**(5), 359–375 (2002)
3. Laporte, G., Semet, F.: Classical heuristics for the capacitated VRP. In: Toth, P., Vigo, D. (eds.) The Vehicle Routing Problem, pp. 109–128. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia (2002)
4. Gendreau, M., Laporte, M.G., Potvin, J.-Y.: Metaheuristics for the capacitated VRP. In: Toth, P., Vigo, D. (eds.) The Vehicle Routing Problem, pp. 129–154. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia (2002)
5. Tarantilis, C.D., Gendreau, M., Spinellis, D.: The Impact of Metaheuristic Strategies in Transportation and Logistics Applications. IEEE Intelligent Syst. **20**(4), 16–18 (2005)

6. Kirkpatrick, S., Gelatt, C.D., Jr., Vecchi, M.P.: Optimization by simulated annealing. Science **220**, 671–680 (1983)
7. Osman, I.H.: Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. Ann. Oper. Res. **41**, 421–451 (1993)
8. Dueck, G., Scheuer, T.: Threshold accepting. A general purpose optimization algorithm appearing superior to simulated annealing. J. Comput. Phys. **90**, 161–175 (1990)
9. Tarantilis, C.D., Kiranoudis, C.T., Vassiliadis, V.S.: A backtracking adaptive threshold accepting metaheuristic method for the Vehicle Routing Problem. Syst. Anal. Model. Simul. **42**(5), 631–644 (2002)
10. Tarantilis, C.D., Kiranoudis, C.T., Vassiliadis, V.S.: A list based threshold accepting algorithm for the capacitated vehicle routing problem. Int. J. Comput. Math. **79**(5), 537–553 (2002)
11. Dueck, G.: New optimization heuristics: the great deluge algorithm and the record to record travel. J. Comput. Phys. **104**, 86–92 (1993)
12. Golden, B.L., Wasil, E.A., Kelly, J.P., Chao, I.-M.: The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets, and computational results. In: Crainic, T., Laporte, G. (eds.) Fleet management and logistics, pp. 33–56. Kluwer, Boston (1998)
13. Li, F., Golden, B.L., Wasil, E.A.: Very large-scale vehicle routing: New test problems, algorithms, and results. In: Computers & Operations Research, Forthcoming (2004)
14. Christofides, N., Mingozzi, A., Toth, P.: The vehicle routing problem. In: Christofides, N., Mingozzi, A., Toth, P., Sandi, C. (eds.) Combinatorial Optimization, pp. 315–338. Wiley, Chichester (1979)
15. Tarantilis, C.D., Kiranoudis, C.T.: Using the Vehicle Routing Problem for the Transportation of Hazardous Materials. Oper. Res. An. Int. J. **1**(1), 67–78 (2001)

R*-Tree

HANS-PETER KRIEGEL, PETER KUNATH, MATTHIAS RENZ
Department of Computer Science, Ludwig-Maximilians-University of Munich, Munich, Germany

Synonyms

Spatial index structure; Spatial access method; R-tree; TPR-trees; Point query; Window query; Range query; Directory rectangles; Storage utilization; Reinsert, forced

Definition

The R*-tree, an improvement of the R-tree, is one of the most popular access methods for points and rectangles. This is achieved by modifying the insert and split algorithms of the original R-tree which is based on the heuristic optimization of the area of the enclosing rectangle in each inner node. The R*-tree incorporates a combined optimization of area, margin and overlap of each enclosing rectangle in the directory. From a practical point of view, the R*-tree is very attractive because of the following two reasons. It efficiently supports point and spatial data at the same time and its implementation cost is only slightly higher than that of other R-tree variants.

Historical Background

The R*-tree, proposed in 1990, is one of the most prominent representatives of the R-tree family. One major reason for using R-tree based index structures is the requirement to index not only point data but also extended spatial data, and R-tree-based index structures are well suited for both types of data. In contrast to most other index structures (such as kdB-trees [1], grid files [2], and their variants, see e. g., [3]), R-tree-based index structures do not need point transformations to store spatial data and, therefore, provide better spatial clustering. The R*-tree differs from the R-tree in its split strategy which is based on a heuristic optimization. It constitutes the foundation of several modern spatial access methods including the X-tree [4] or the TPR-tree [5], specialized access methods that cope with high-dimensional data and moving objects.

Scientific Fundamentals

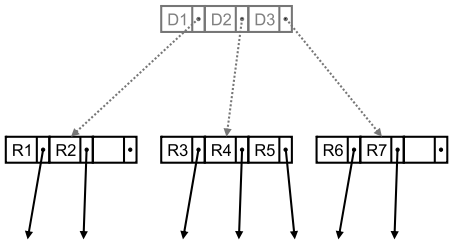
The R*-tree proposed in [6] is an extension of the R-tree which was designed to overcome certain drawbacks of the R-tree. An example of the R*-tree structure and the data space with the corresponding bounding rectangles is depicted in Fig. 1.

Optimization Criteria for R-Trees

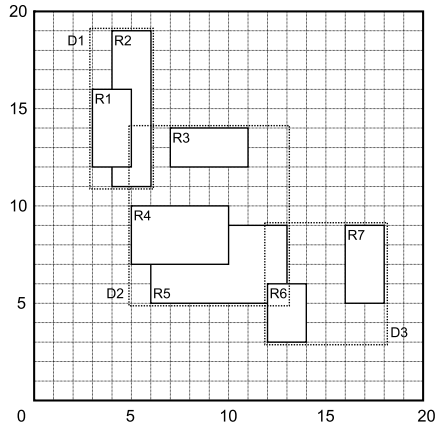
For a set of given data objects, an R-tree dynamically constructs bounding boxes from subsets between m and M rectangles. This is done in a way that efficiently supports retrieval operations such as point- and window-queries of arbitrary size. The known parameters of good retrieval performances affect each other in a very complex way, such that it is impossible to optimize one of them without influencing other parameters. This may then lead to a deterioration of the overall performance. Moreover, since the data rectangles may have very different size and shape, and the directory rectangles grow and shrink dynamically, the success of methods which optimize only one parameter is very unlikely.

In the following text, some of the parameters which are essential for the retrieval performance are considered. Furthermore, interdependencies between different parameters and optimization criteria are analyzed.

(O1) The area covered by a directory rectangle should be minimized, i. e., the area covered by the bounding rectangle, but not covered by the enclosed rectangles,



Structure of the R*-tree with inner nodes (light color) and leaf nodes (dark color).



Data space with directory rectangles D1 – D3 (dashed lines) and data rectangles R1 – R7 (solid lines).

R*-Tree, Figure 1 R*-tree example

should be minimized. This area is often referred to as *dead space*. This improves the performance since decisions regarding which paths have to be traversed can be made on higher levels.

(O2) The overlap between directory rectangles should be minimized. This also decreases the number of paths to be traversed.

(O3) The margin of a directory rectangle should be minimized. Here, the margin is the sum of the lengths of the edges of a rectangle. Assuming a fixed area, the rectangular object with the smallest margin is a square. Thus, minimizing the margin instead of the area will result in directory rectangles that are more quadratic shaped. Queries with large quadratic query rectangles especially profit from this optimization. More importantly, the minimization of the margin improves the overall structure. Since quadratic objects can be packed easier, the bounding boxes of a level will form smaller directory rectangles on the level above. Thus, clustering rectangles into bounding boxes with only little variance of the length of the edges will reduce the area of the directory rectangles.

(O4) The storage utilization should be optimized. Higher storage utilization generally reduces the query cost as the height of the tree is kept low. Queries with a low selectivity especially benefit from a good storage utilization because the number of visited nodes for such queries is usually higher than that for queries with a high selectivity.

Unfortunately, these four criteria are not necessarily in accordance with each other. Keeping the area and overlap of a directory rectangle small requires more freedom in the number of rectangles stored in one node. Minimizing these parameters usually comes at the cost of decreasing the storage utilization. Moreover, when fulfilling (O1) or (O2), more freedom in choosing the shape is necessary. Thus, the rectangles will be less quadratic. When focusing on (O1), the overlap between directory rectangles may be affected in a positive way since the data space covered by the directory rectangles, in particular the *dead space*, is reduced. As for every geometric optimization, minimizing the margins also leads to a reduced storage utilization. However, since a more quadratic shape of the directory rectangles results in better packing, it is easier to maintain high storage utilization. Obviously, the performance of queries with sufficiently low selectivity will be affected more by the storage utilization than by the parameters of (O1)–(O3).

Choosing the Subtree

To solve the problem of choosing an appropriate insertion path, previous R-tree versions take only the area into consideration. The authors of the R*-tree tested the parameters area, margin, and overlap in different combinations. The overlap of an entry is defined as follows. Let $E_1, \ldots, E_p$ be the entries in the current node. Then,

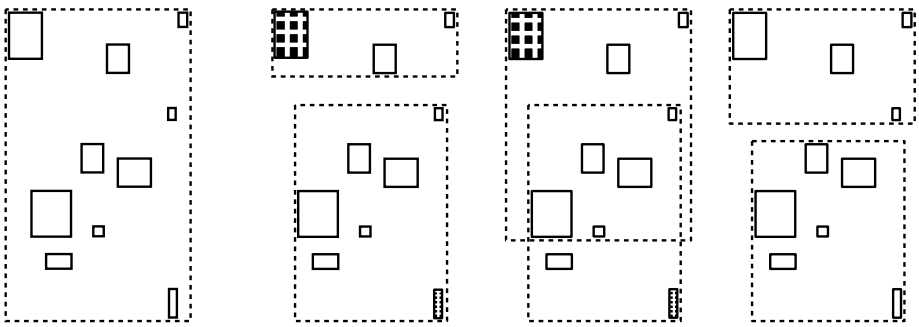
$$\text{overlap}(E_k) = \sum_{i=1, i \neq k}^{p} \text{area}(E_i \cap E_k), 1 < k < p.$$

Figure 2 describes the algorithm which performed best in the experimental evaluation conducted by the R*-tree authors. For choosing the best non-leaf node, alternative methods did not outperform Guttman's original algorithm. For the leaf nodes, minimizing the overlap performed slightly better.

In this version, the CPU cost to determine the overlap are quadratic in the number of node entries, because for each entry, the overlap with all other entries of the node has to be calculated. However, for large node sizes, one can reduce

```
ChooseSubtree(N: Node)
CS1  Choose N to be the root
CS2  IF N is a leaf
        return N
     ELSE
        IF the childpointers in N point to leaves
           // Determine minimum overlap cost
           Choose the entry in N whose rectangle needs
           least overlap enlargement to include the new data.
           Resolve ties by choosing the entry whose rectangle
           needs least area enlargement.
           If there is still a tie, choose the entry with
           the rectangle that has the smallest area.
        ELSE // the childpointers in N do not point to leaves
           // Determine minimum area cost
           Choose the entry in N whose rectangle needs least
           area enlargement to include the new data rectangle.
           Resolve ties by choosing the entry with the
           rectangle that has the smallest area.
        END
     END
CS3     Set N to be the childnode pointed to by the
        childpointer of the chosen entry and repeat from CS2
```

R*-Tree, Figure 2 The algorithm for finding the most appropriate subtree



Overfilled node | Split of the quadratic R-tree, m = 30% | Split of the quadratic R-tree, m = 40% | Split of the R*-tree, m = 40%

R*-Tree, Figure 3 Different partitioning variants

the number of entries for which the calculation has to be done. The idea is that for very distant rectangles, the probability to yield the minimum overlap is very small. Thus, in order to reduce the CPU cost, the first part of CS2 can be modified as follows:

```
// Determine the minimum overlap cost (fast version)
Sort the rectangles in node N in increasing order of their
area enlargement needed to include the new data
rectangle. Let A be the group of the first p entries.
From the entries in A, consider all entries in N and
choose the entry whose rectangle needs least overlap
enlargement. Resolve ties as described above.
```

The R*-tree authors showed that this modified version leads to almost no reduction of retrieval performance when using two-dimensional data and setting the p parameter to 32. Nevertheless, the CPU costs remain higher than in the R-tree version of `ChooseSubtree`. However, the number of disk accesses is reduced for the exact match query preceding each insertion and is reduced for the `ChooseSubtree` algorithm itself. In particular, highly selective window queries on data sets consisting of non-uniformly distributed small rectangles or points benefit from the `ChooseSubtree` optimization. For other data sets, the performance of Guttman's algorithm is usually similar to this one.

Splitting a Node

For determining a good split of a directory node, it is necessary to find an adequate partitioning of the MBRs in the node into two subsets. Figure 3 illustrates typical problems that arise when using an inappropriate partitioning of the entries of a node. The example shows that the split strategy of the R*-tree leads to a more effective partitioning than that of the R-tree. The depicted split results are based on the quadratic split of the R-tree for a varying mini-

Split
S1 Invoke ChooseSplitAxis to determine the axis perpendicular to which the split is performed.
S2 Invoke ChooseSplitIndex to determine the best distribution into two groups along that axis.
S3 Distribute the entries into two groups.

ChooseSplitAxis
CSA1 For each axis: Sort the entries by the lower value, then by the upper value of their rectangles and determine all distributions.
Compute S, the sum of all margin-values of the different distributions.
CSA2 Choose the axis with the minimum S as the split axis.

ChooseSplitIndex
CSI1 Along the chosen split axis, choose the distribution with the minimum overlap-value. Resolve ties by choosing the distribution having the minimum area-value.

R*-Tree, Figure 4 The R*-tree Split algorithm

mum number of entries m relative to M. The result is either a split with uneven distribution of the entries, reducing the storage utilization (first partitioning), or a split with much overlap (second partitioning). For comparison, the third partitioning illustrates the results from the R*-tree split.

The R*-tree uses the following method to find good splits which is depicted in Fig. 4.

In a first step (S1), the split axis has to be chosen. This is carried out using the function *ChooseSplitAxis* in the following way. Along each axis, the entries are first sorted by the lower value of their rectangles, then sorted by the upper value of their rectangles. For each sort $M-2m+2$ distributions of the $M+1$ entries into two groups are determined, where the k-th distribution $(k=1,\ldots,(M-2m+2))$ is determined as follows. The first group contains the first $(m-1)+k$ entries, the second group contains the remaining entries. For each of the M–$2m+2$ distributions, the *margin-value* is determined by summarizing the margin length of the two minimum bounding boxes of both distributions. Finally, the axis which yields the minimum *margin-value* is chosen as a split axis.

In the next step (S2), an adequate partitioning of the entries along the split axis determined in the previous step has to be found. This is done by the function *ChooseSplitIndex*. This time, the *overlap-value* for each of the $2\cdot(M-2m+2)$ distributions is considered where the *overlap-value* denotes the size of the area of the overlap between the two minimum bounding boxes of both partitions. Here, ties are resolved by choosing the distribution with the minimum *area-value* denoting the sum of the size of the areas covered by both minimum bounding boxes of both partitions.

Note that each of the parameters, *margin-value*, *overlap-value* and *area-value*, potentially may be chosen for the determination of the split axis and the final distribution in an arbitrary sequence. In the present experiments, the proposed constellation, i. e., *margin-value* for the determination of the split axis and *overlap-value* and *area-value* for the final choice of the partitioning, has shown the best overall performance.

As experiments with several values of M have shown, $m = 40\%$ yields the best performance. For each axis (dimension), the entries have to be sorted twice, requiring $O(M \log M)$ time. As an experimental cost analysis showed, this needs about half of the cost of the split. The remaining split cost is spent computing the margin of the $2\cdot(M-2m+2)$ distributions.

Forced Reinsert

Both the R-tree and the R*-tree are nondeterministic in allocating the entries to the nodes, i. e., different sequences of insertions will generate different trees. Obviously, the retrieval performance of the R-tree can suffer from its old entries. Data rectangles inserted during the early growth of the structure may have introduced directory rectangles which are not suitable to guarantee a good retrieval performance in the current situation. A very local reorganization of the directory rectangles is performed during a split. However, this is rather poor and it is thus desirable to have a more powerful and less local instrument to reorganize the structure.

The discussed problem would be maintained or even worsened if underfilled nodes, resulting from the deletion of records, would be merged under the old parent. Thus, the approach of treating underfilled nodes in an R-tree is to delete the node and to reinsert the orphaned entries in the corresponding level [7]. This way, the ChooseSubtree algorithm has a new chance of distributing entries into different nodes. Therefore, randomly deleting parts of the data and then reinserting it seems to be a very simple way of tuning existing R-tree data files. However, this is a static

InsertData
ID1 Invoke Insert starting with the leaf level as a parameter to insert a new data rectangle.

Insert
I1 Invoke ChooseSubtree with the level as a parameter to find an appropriate node N, in which to place the new entry E.
I2 If N has less then M entries, accommodate E in N.
If N has M entries, invoke OverflowTreatment with the level of N as a parameter (for reinsertion or split).
I3 If OverflowTreatment was called and a split was performed, propagate OverflowTreatment upwards if necessary.
If OverflowTreatment caused a split of the root, create a new root.
I4 Adjust all covering rectangles in the insertion path such that they are minimum bounding boxes enclosing their children rectangles.

OverflowTreatment
OT1 If the level is not the root level and this is the first call of OverflowTreatment in the given level during the insertion of one data rectangle, then invole Reinsert, else invoke Split.

Reinsert
RI1 For all $M + 1$ entries of a node N compute the distance between the centers of their rectangles and the center of the bounding rectangle of N.
RI2 Sort the entries in decreasing order of their distances computed in RI1.
RI3 Remove the first p entries from N and adjust the bounding rectangle of N.
RI4 In the sort defined in RI2, starting with the maximum distance (= far reinsert) or minimum distance (= close reinsert), invoke Insert to reinsert the entries.

R*-Tree, Figure 5 The R*-tree Reinsert algorithm

situation, and for nearly static data files, the pack algorithm [8] is a more sophisticated approach.

In order to achieve dynamic reorganizations, the R*-tree forces entries to be reinserted during the insertion routine. The algorithm `Reinsert` in Fig. 5 is based on the ability of the insert routine to insert entries on every level of the tree, as already required by the deletion algorithm [7].

If a new data rectangle is inserted, the first overflow treatment on each level will be reinsertion of p entries, where p is a percentage value to be optimized. This may cause a split in the node which caused the overflow if all entries are reinserted in the same location. Otherwise, splits may occur in one or more of the other nodes, however, in many situations splits are completely prevented. The parameter p can be varied independently for leaf nodes and non-leaf nodes as part of performance tuning. Experiments have shown that $p = 30\%$ of M for leaf nodes as well as non-leaf nodes yield the best performance. Furthermore, experiments demonstrated that close Reinsert outperforms far Reinsert. Close reinsert prefers the node which included the entries before, and this is intended, because its enclosing rectangle was reduced in size. Thus, this node has lower probability to be selected by ChooseSubtree again.

In summary, the following properties apply to the concept of Forced Reinsert.

- Forced Reinsert relocates entries between neighboring nodes and thus decreases the overlap.
- As a side effect, storage utilization is improved.
- Due to more restructuring, less splits occur.
- Since the outer rectangles of a node are reinserted, the shape of the directory rectangles will be more quadratic.
- Higher CPU cost due to the need to call the insertion routine more often is alleviated by having to perform less splits.
- Experimental evaluations show that even with Forced Reinsert, the average insertion cost of the R*-tree is lower than that for the other R-tree variants.

Key Applications

A spatial object is regarded as a distinct entity occupying an individual location in a one- or multidimensional data space. Furthermore, it may be extended along some (or all) dimensions. A complex spatial object of the real world can be considered as a collection of individual, two- or three-dimensional components, where each component

potentially represents a complex and intricate geometric shape. Examples of such complex objects are geographical regions, or parts of a car or an airplane.

Geographic Information Systems (GIS)

Geographic Information Systems, abbreviated as GIS, are computer-based systems that have been developed and designed specifically to handle geographic information. They are designed to allow the user to capture spatial information, its storage, analysis, manipulation and the production of maps as outputs. Since spatial information can be efficiently handled by means of spatial access methods like the R*-tree, these systems are able to hold an enormous range and quantity of information.

Digital Mock-up (DMU)

In CAD databases, each instance of a part occupies a specific region in the two- or three-dimensional product space. Together, all parts of a given product version and its variants represent a virtual prototype of the constructed geometry. One of the most important tasks for DMU are collision detection and proximity queries that can be efficiently supported by the R*-tree.

Multidimensional Feature Vectors

While data sets usually are organized with respect to primary keys, the remaining entries often resemble the interesting part of the data set. A set of numerical (and also certain categorical) features often fulfills the properties of vector spaces, and thus, database objects can be considered as points in space. The R*-tree is one of the most frequently used access methods for objects in multidimensional vector spaces. It can manage point objects as well as spatially extended objects. Due to its good clustering properties, it is very suitable for proximity queries like distance range queries and nearest neighbor queries that are often used for similarity search.

Future Directions

The R*-tree can be efficiently used as an access method in database systems organizing both multidimensional point data and spatial data. Since modern access methods have to cope with more complex object representations such as high-dimensional data, dynamic objects, or uncertain data, variants or extensions of the R*-tree may be more suitable. Examples of such variants are the X-tree (for high-dimensional data), the TPR-tree (for dynamic objects), and the Gauss-tree (for uncertain data).

Cross References

- ▶ Indexing, High Dimensional
- ▶ Indexing, Hilbert R-tree, Spatial Indexing, Multimedia Indexing
- ▶ Indexing, X-Tree
- ▶ R-Trees – A Dynamic Index Structure for Spatial Searching
- ▶ Vector Data

Recommended Reading

1. Robinson, J.T.: The K-D-B-tree: a search structure for large multidimensional dynamic indexes. In: Proc. of the 1981 ACM SIGMOD international conference on Management of data (SIGMOD'81), Ann Arbor, Michigan (1981)
2. Nievergelt, J., Hinterberger, H., Sevcik, K.C.: The grid file: An adaptable, symmetric multikey file structure. ACM Trans Database Syst **9**(1):38–71 (1984)
3. Seeger, B., Kriegel, H.-P.: The buddy tree: an efficient and robust access method for spatial data bases. In: Proceedings of 16th international conference on Very large databases (VLDB'90), Brisbane, Australia (1990) pp. 590–601
4. Berchtold, S., Keim, D.A., Kriegel, H.-P: The X-tree: An index structure for high-dimensional data. In: Proc. 22nd Int. Conf. on Very Large Data Bases (VLDB'96), Bombay, India (1996) pp. 28–39
5. Saltenis, S., Jensen, C.S., Leutenegger, S.T., Lopez, M.A.: Indexing the positions of continuously moving objects. In: SIGMOD Conference, (2000) pp. 331–342
6. Beckmann, N., Kriegel, H.-P., Schneider, R., Seeger, B.: The R*-tree: An efficient and robust access method for points and rectangles. In: Proceedings of the International Conference on Management of Data (SIGMOD'90), Atlantic City, NJ (1990) pp. 322–331
7. Guttman, A.: R-trees: a dynamic index structure for spatial searching. In: Proc. of the 1984 ACM SIGMOD international conference on Management of data (SIGMOD'84), Boston, Massachusetts (1984) pp. 47–57
8. Roussopoulos, N., Leifker, D.: Direct spatial search on pictorial databases using packed R-trees. In: Proc. of the 1985 ACM SIGMOD international conference on Management of data (SIGMOD'85), Austin, Texas (1985) pp. 17–31

R-Tree

- ▶ Indexing Schemes for Multi-dimensional Moving Objects
- ▶ Indexing, X-Tree
- ▶ PostGIS
- ▶ R*-tree
- ▶ R-Trees – A Dynamic Index Structure for Spatial Searching

R-Tree, Multi-Version

- ▶ Movement Patterns in Spatio-temporal Data

R-Trees – A Dynamic Index Structure for Spatial Searching

MARIOS HADJIELEFTHERIOU[1],
YANNIS MANOLOPOULOS[2], YANNIS THEODORIDIS[3],
VASSILIS J. TSOTRAS[4]
[1] AT&T Labs, Inc., Florham Park, NJ, USA
[2] Department of Informatics, Aristotle University, Thessaloniki, Greece
[3] Department of Informatics, University of Piraeus, Piraeus, Greece
[4] University of California-Riverside, Riverside, CA, USA

Synonyms

R-tree

Definition

One of the most influential access methods in the area of Spatial Data Management is the R-tree structure proposed by Guttman in 1984 [8]. It is a hierarchical data structure based on B^+-trees, used for the dynamic organization of a set of d-dimensional geometric objects. The original R-tree was designed for efficiently retrieving geometric objects contained within a given query range. Every object in the R-tree is represented by a minimum bounding d-dimensional rectangle (for simplicity, MBRs in the sequel). Data objects are grouped into larger MBRs forming the *leaf nodes* of the tree. Leaf nodes are grouped into larger *internal nodes*. The process continues recursively until the last group of nodes that form the root of the tree. The root represents an MBR that encloses all objects and nodes indexed by the tree, and each node corresponds to the MBR that bounds its children (cf. Fig. 1). A range query can be answered efficiently by traversing the tree starting from the root and ending at the leaves, accessing only nodes whose MBRs intersect with the query range. At the leaf level of the tree, the actual geometric objects are retrieved and tested for true containment in the query. Several variations of the original structure have been proposed to provide more efficient access, handle objects in high-dimensional spaces, support concurrent accesses, support I/O and CPU parallelism, support efficient bulk loading, and several other types of spatial and spatio-temporal queries, like nearest neighbors, spatial joins, and more.

Historical Background

The 1980s were a period of wide acceptance of relational systems in the market, but at the same time it became apparent that the relational model was not adequate to host new emerging applications. Multimedia, CAD/CAM, geographical, medical and scientific applications are just some examples, in which the relational model had been proven to behave poorly. Thus, the object-oriented model and the object-relational model were proposed. One of the reasons for the shortcoming of the relational systems was their inability to handle the new kinds of data with B-trees. More specifically, B-trees were designed to handle alphanumeric (i. e., one-dimensional) data, like integers, characters, and strings, where an ordering relation can be defined. In light of this development, entirely novel access methods were proposed, evaluated, compared, and established. One of these structures, the R-tree, was proposed by Guttman in 1984, aimed to handle geometrical data, such as points, line segments, surfaces, volumes, and hyper volumes in high-dimensional spaces [8]. R-trees were treated in the literature in much the same way as B-trees. In particular, many improving variations have been proposed for various instances and environments, several novel operations have been developed, and new cost models have been suggested.

It seems that due to modern demanding applications and after academia has paved the way, the industry recently recognized the use and necessity of R-trees. Thus, R-trees are adopted as an additional access method to handle multi-dimensional data. Nowadays, spatial databases and geographical information systems have been established as a mature field, spatio-temporal databases and manipulation of moving points and trajectories are being studied extensively, and finally image and multimedia databases able to handle new kinds of data, such as images, voice, music, or video, are being designed and developed. An application in all these cases should rely on R-trees as a necessary tool for data storage and retrieval. R-tree applications cover a wide spectrum, from spatial and temporal to image and video (multimedia) databases. The initial application that motivated Guttman in his pioneering research was VLSI design (i. e., how to efficiently answer whether a space is already covered by a chip). Gradually, handling rectangles quickly found applications in geographical and, in general, spatial data, including GIS (buildings, rivers, cities, etc.), image or video/audio retrieval systems (similarity of objects in either original space or high-dimensional feature space), time series and chronological databases (time intervals are just 1D objects), and so on [18].

Scientific Fundamentals

The Original R-Tree

The R-tree comprises a generalization of the B^+-tree structure for multiple dimensions. An R-tree of order (m, M) has the following characteristics:

1. The root node of the tree contains at least two entries, unless it is a leaf (in this case, it may contain zero or a single entry).
2. Internal nodes can store between $m \leq M/2$ and M child entries. Each entry is of the form (*p, mbr*), where *p* is a pointer to a children node and *mbr* is the MBR that spatially encloses all entries contained in the sub-tree rooted at this child.
3. Each leaf node (unless it is the root) can store between $m \leq M/2$ and M entries. Each entry is of the form *oid*, *mbr*), where *oid* is an object identifier and *mbr* is the MBR that spatially encloses this object.
4. The R-tree is a height-balanced structure, i. e., all leaves appear at the same level of the tree.

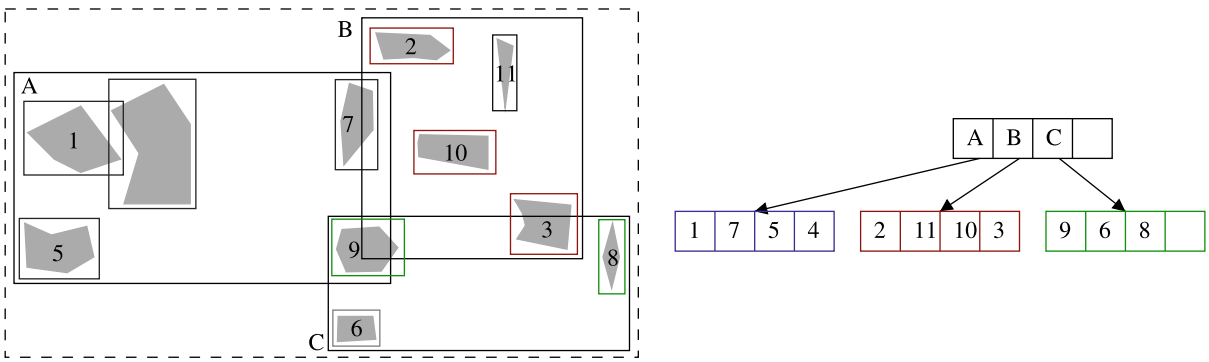
Let an R-tree store N data objects. The maximum possible height h of the tree is $h_{\max} = \lceil \log_m N \rceil - 1$. The maximum number of nodes (by assuming that each node contains the minimum allowed number of entries) is $\sum_{i=1}^{h_{\max}} \lceil N/m^i \rceil = \lceil N/m \rceil + \lceil N/m^2 \rceil + \ldots + 1$.

Figure 1 shows a set of geometric objects, their enclosing MBRs, and a grouping into larger MBRs forming the index levels of the R-tree (assuming that $M = 4$ and $m = 2$). The actual hierarchy of the R-tree is shown on the right. It is evident that several R-trees can represent the same set of data rectangles, depending on the grouping of data objects into leaves, and internal nodes into larger nodes. Hence, which R-tree is constructed is determined by the insertion and/or deletion algorithm used.

Notice also that node MBRs on the same level of the tree might overlap each other. Besides, in a geometric sense a given MBRs may be contained in multiple nodes, but from an R-tree perspective it is associated with only one node of the tree. This means that a spatial search may have to traverse multiple paths of the tree before confirming the existence of a given MBR. Also, the representation of geometric objects with MBRs may result in false alarms. To resolve false alarms, the actual objects must be examined. Therefore, the R-tree plays the role of a filtering mechanism to reduce the costly direct examination of geometric objects, and limit the search space.

Given a query rectangle Q *a range search or a window query* (i. e., retrieving all data objects that intersect with Q) is performed by traversing the tree starting from the root and ending at the leaves, retrieving only nodes that intersect with Q. The algorithm is shown in Fig. 2. For a node entry e, $e.mbr$ denotes the child MBR and $e.p$ the child pointer to the next level. If the node is a leaf, then $e.p$ is the object identifier (*oid*).

Note that the objects returned by the RangeSearch procedure are potential candidate answers. The actual geometric



R-Trees – A Dynamic Index Structure for Spatial Searching, Figure 1 The hierarchy of an R-tree. Data objects are represented by their enclosing MBRs, which are grouped into larger nodes hierarchically until the root node of the tree

Algorithm RangeSearch (TypeNode RN**, TypeRegion** Q**)**
/* Finds all rectangles that are stored in an R-tree with root node RN, which intersect with a query rectangle Q. Answers are stored in set $\mathcal{A}$ */

1. **if** RN is not a leaf node
2. examine each entry e of RN to find $e.mbr$ that intersect Q
3. **foreach** such entry e call RangeSearch($e.p$,Q)
4. **else** // RN is a leaf node
5. examine each entry e to find $e.mbr$ that intersects Q
6. add these entries to answer set $\mathcal{A}$
7. **endif**

R-Trees – A Dynamic Index Structure for Spatial Searching, Figure 2 The R-tree range search algorithm

Algorithm Insert (TypeEntry E**, TypeNode** RN**)**
/* Inserts a new entry E in an R-tree with root node RN */

1. Traverse the tree from root RN to the appropriate leaf. At each level, select the node, L, whose MBR will require the minimum area enlargement to cover $E.mbr$
2. In case of ties, select the node whose MBR has the minimum area
3. **if** the selected leaf L can accommodate E
4. Insert E into L
5. Update all MBRs in the path from the root to L, so that all of them cover $E.mbr$
6. **else** // L is already full
7. Let $\mathcal{E}$ be the set consisting of all L's entries and the new entry E
   Select as seeds two entries $e_1, e_2 \in \mathcal{E}$, where the distance between e_1 and e_2 is the maximum among all other pairs of entries from $\mathcal{E}$
   Form two nodes, L_1 and L_2, where the first contains e_1 and the second e_2
8. Examine the remaining members of $\mathcal{E}$ one by one and assign them to L_1 or L_2, depending on which of the MBRs of these nodes will require the minimum area enlargement so as to cover this entry
9. **if** a tie occurs
10. Assign the entry to the node whose MBR has the smaller area
11. **endif**
12. **if** a tie occurs again
13. Assign the entry to the node that contains the smaller number of entries
14. **endif**
15. **if** during the assignment of entries, there remain λ entries to be assigned and the one node contains $m - \lambda$ entries
16. Assign all the remaining entries to this node without considering the aforementioned criteria
    /* so that the node will contain at least m entries */
17. **endif**
18. Update the MBRs of nodes that are in the path from root to L, so as to cover L_1 and accommodate L_2
19. Perform splits at the upper levels if necessary
20. In case the root has to be split, create a new root
21. Increase the height of the tree by one
22. **endif**

R-Trees – A Dynamic Index Structure for Spatial Searching, Figure 3 The R-tree insertion algorithm

objects intersected by the query rectangle have to be identified using a refinement step by retrieving the objects and testing for true intersection.

R-trees are dynamic data structures, i. e., global reorganization is not required to handle insertions or deletions. Insertions are handled similarly to insertions in a B^+-tree. In particular, the R-tree is traversed to locate an appropriate leaf to accommodate the new entry. The entry is inserted in that leaf and all nodes in the path from the leaf to the root are updated accordingly. In case the leaf cannot accommodate the new entry because it already contains M entries, it is split into two new nodes. The algorithm for inserting a new data rectangle in an R-tree is presented in Fig. 3.

Splitting in R-trees differs from that of the B^+-tree. Given that R-tree nodes are allowed to overlap, the objective of a split algorithm is to minimize the probability of retrieving both new nodes (L_1 and L_2) for the same query. The splitting criteria introduced by Guttman are the following:

Linear Split. Choose the two objects lying the furthest apart from each other as seeds for the two new nodes. Then consider each remaining object in a random order and assign it to the node requiring the smallest enlargement of its respective MBR. The linear split algorithm tries to minimizing the total area of the two new nodes.

Quadratic Split. Choose two objects as seeds for the two nodes. The seeds, if grouped together create as much dead space as possible (*dead space* is the area covered by the MBR but not by the enclosed children). Then, until there are no remaining objects, choose the object

Algorithm Delete (TypeEntry E**, TypeNode** RN**)**
/* Deletes an entry E from an R-tree with root node RN */

```
1.  if RN is a leaf node
2.      search all entries of RN to find E.mbr
3.  else // RN is an internal node
4.      Find all entries of RN that cover E.mbr
5.      Follow the corresponding subtrees until the leaf L that contains E is found
6.      Remove E from L
7.  endif
8.  Call algorithm CondenseTree(L) /* Figure 5 */
9.  if the root has only one child /* and it is not a leaf */
10.     Remove the root
11.     Set as new root its only child
12. endif
```

R-Trees – A Dynamic Index Structure for Spatial Searching, Figure 4 The R-tree deletion algorithm

that maximizes the difference of dead space if assigned to each of the two nodes, and insert it in the node that requires the least enlargement of its MBR.

Exponential Split. All possible groupings are exhaustively tested and the best is chosen with respect to the minimization of the MBR enlargement.

Regarding the deletion of an entry from an R-tree, it is performed with the algorithm given in Fig. 4. Notice that the R-tree is using re-insertions instead of merging sibling nodes to handle entries contained in nodes that underflow (i. e., nodes that end up with less than m children). Reinsertion achieves the same result as merging. Additionally, the algorithm for insertion is reused. Also, most of the nodes accessed during reinsertion are available in buffer memory, because they were retrieved during searching for the deleted entry. The standard deletion technique is compared with alternative techniques in [19].

R-Tree Variants

The original R-tree has two important disadvantages that motivated the study of more efficient variations:

1. The execution of a point location query in an R-tree may lead to the investigation of several paths from the root to the leaf level. This characteristic may lead to performance deterioration, specifically when the overlap of the MBRs is significant.
2. A few large rectangles may increase the degree of overlap significantly, leading to performance degradation during range query execution, due to empty space.

The R$^+$-Tree R$^+$-trees [23] were proposed as a structure that avoids visiting multiple paths during point location queries, and thus leading to improved retrieval performance. The R$^+$-tree avoids MBR overlapping of internal nodes at the same level of the tree, by using the clipping technique. Inserted objects and nodes are divided in two or more MBRs if needed. This means that a specific object *oid* may be duplicated and stored redundantly in several nodes.

The algorithm for range query processing is similar to the one used for R-trees. The only difference is that duplicate elimination is necessary to avoid reporting an object more than once. However, insertion, deletion, and node splitting algorithms are different due to the clipping technique. In order to insert a new entry E, the insertion algorithm starts from the root and determines the MBRs that intersect $E.mbr$. Then $E.mbr$ is clipped and the procedure is recursively applied for the corresponding sub-trees. During the execution of the insertion algorithm a node may become full. To handle this situation, a node splitting mechanism is required as in the R-tree case. The main difference between the R$^+$-tree splitting algorithm and that of the R-tree is that downward propagation may be necessary due to object redundancy, in addition to the upward propagation. At the same time, another side effect of clipping is that during insertions, an MBR augmentation may lead to a series of update operations in a chain reaction type. Also, under certain circumstances, the structure may lead to a deadlock, as, for example, when a split has to take place at a node with M+1 rectangles, where every rectangle is enclosed into another. The fact that multiple copies of an object's MBR may be stored in several leaf nodes has a direct impact on the deletion algorithm. All copies of an object's MBR must be removed from the corresponding leaf nodes. Evidently, an increased number of deletions may reduce storage utilization significantly. Therefore, appropriate reorganization must be performed to handle underutilized tree nodes as well.

The R*-Tree R*-trees [2] are widely accepted in the literature as a prevailing performance-wise structure that is

Algorithm CondenseTree (TypeNode L)
/* Given is the leaf L from which an entry E has been deleted, if after the deletion of E, L has fewer than m entries, remove entirely leaf L and reinsert all remaining entries. Updates are propagated upwards and the MBRs in the path from the root to L are modified (possibly become smaller) */

1. Set $X = L$
2. Let $\mathcal{N}$ be the set of nodes that are going to be removed from the tree (initially, $\mathcal{N}$ is empty)
3. **while** X is not the root
4. Let $Parent_X$ be the father node of X
5. Let E_X be the entry of $Parent_X$ that corresponds to X
6. **if** X contains less than m entries
7. Remove E_X from $Parent_X$
8. Insert X into $\mathcal{N}$
9. **endif**
10. **if** X has not been removed
11. Adjust its corresponding MBR $E_X.mbr$, so as to enclose all rectangles in X /* $E_X.mbr$ may become smaller */
12. **endif**
13. Set $X = Parent_X$
14. **endwhile**
15. Reinsert all the entries of nodes that are in the set $\mathcal{N}$

R-Trees – A Dynamic Index Structure for Spatial Searching, Figure 5 The R-tree condense algorithm

often used as a basis for performance comparisons. As already discussed, the R-tree is based solely on the area minimization of node MBRs during splitting. On the other hand, the R*-tree examines several other splitting criteria, which intuitively are expected to improve the performance during query processing. The criteria considered are the following:

Minimization of the area covered by each MBR. This criterion aims to minimize the dead space and consequently reduce the number of paths that need to be traversed during query processing. This is the single criterion that is also examined by the R-tree.

Minimization of overlap between MBRs. The larger the overlapping, the larger the expected number of paths traversed by a query.

Minimization of MBR margins (perimeters). Minimization of the perimeter has a direct effect on the shape of the MBR. Regular quadrilaterals (squares) have the smallest perimeter between rectangles with equal area (as opposed to elongated rectangles). Regular shapes help group MBRs more compactly and reduce empty space.

Maximization of storage utilization. When utilization is low, more nodes tend to be examined during query processing, especially for larger queries where a significant portion of the entries intersect with the query. Moreover, the tree height increases with decreasing node utilization.

The R*-tree follows an engineering approach to find the best possible combinations of the aforementioned criteria. This heuristic approach is necessary, because the criteria can become contradictory. For instance, to keep both the area and overlap low, one can reduce the minimum number of entries allowed within a node. Although, this has direct impact on storage utilization. Also, by minimizing the margins in order to have more regular shapes, node overlapping may be increased.

For the insertion of a new entry e, the R*-tree decides which branch to follow by choosing the entry whose MBR needs the *least area enlargement* to cover *e.mbr*. For leaf nodes, the R*-tree considers node *overlapping minimization* instead. For leaves that are full, the R*-tree does not immediately resort to node splitting. Instead, it picks a fraction of the entries from the chosen leaf and reinserts them. The set of entries to be reinserted are those whose centroid distances from node centroid are among the largest 30% (i. e., this is a heuristic to detect and discard the furthest entries). The reinsertion algorithm achieves a kind of tree re-balancing that significantly improves performance during query processing. However, reinsertion is a costly operation. Therefore, only one application of reinsertion is permitted for each level of the tree. When overflow cannot be handled by reinsertion, node splitting is performed. The split algorithm consists of two steps. The first step decides a split axis among all dimensions. The split axis is the one with the smallest overall perimeter. When the split axis is selected, the split algorithm sorts

the entries (according to their lower or upper boundaries) on the selected dimension and examines all possible divisions. The final division is the one that has the minimum overlap between the MBRs of the resulting nodes. The cost of the split algorithm of the R*-tree is computed as follows. Let M and m be the maximum and minimum allowed number of entries, respectively. The entries are sorted once for every dimension, with cost $O(M \log M)$. For each axis, the margin of 2 x 2 x $(M - 2m + 2)$ rectangles and the overlap of 2 x $(M - 2m + 2)$ divisions is calculated. The R*-tree does not use any specialized deletion algorithm. Instead, deletion in the R*-tree is performed with the deletion algorithm of the original R-tree.

Other Variants The Hilbert R-tree [13] is a hybrid structure. It is a B+-tree with geometrical objects being characterized by the Hilbert value of their centroid. The structure is based on the Hilbert space-filling curve which has been shown to preserve the proximity of spatial objects very well.

Ang and Tan in [1] have proposed a linear algorithm to distribute the objects of an overflowing node in two sets. The primary criterion of this algorithm is to distribute the objects between the two nodes as evenly as possible, whereas the second criterion is the minimization of the overlapping between them. Finally, the third criterion is the minimization of the total coverage.

Garcia et al. [7] proposed a new optimal polynomial splitting algorithm $O(n^d)$, where d is the space dimensionality and $n = M + 1$ is the number of entries of the node that overflows. For n rectangles the number of possible bi-partitions is exponential in n. Each bi-partition is characterized by a pair of MBRs, one for each set of rectangles in each partition. The key issue, however, is that a large number of candidate bi-partitions share the same pair of MBRs. This happens when rectangles that do not participate in the formation of the new enclosing MBRs are exchanged. The authors show that if the cost function used depends only on the characteristics of the MBRs, then the number of different partitionings is polynomial.

More recently, Schreck and Chen [22] proposed an insertion heuristic to improve the shape of the R-tree so that the tree achieves a more elegant shape, with a smaller number of nodes and better storage utilization. In particular, this technique considers how to redistribute data among neighboring nodes, so as to reduce the total number of created nodes. This approach is motivated by the fact that if an overflowing entry could be accommodated by another node the split could be prevented. In this case, a split is performed only when all nodes are completely full.

Huang et al. proposed Compact R-trees, a dynamic R-tree version with optimal space overhead [11]. The motivation behind the proposed approach is that R-trees, R+-trees, and R*-trees suffer from the storage utilization problem, which is around 70% in the average case. Therefore, the authors improve the insertion mechanism of R-trees to a more compact R-tree structure, with no penalty on performance during queries.

Motivated by the analogy between separating of R-tree node entries during the split procedure on the one hand and clustering of spatial objects on the other hand, Brakatsoulas et al. [3], implemented a novel splitting procedure that results in up to k nodes ($k \geq 2$ being a parameter), using the k-means clustering as a working example.

Static R-Trees

There are many applications that index only static data. For instance, insertions and deletions in census, cartographic and environmental databases are rare. For such applications, special attention should be paid to construct an optimal structure with regard to some tree characteristics, such as storage utilization maximization, minimization of overlap or coverage between tree nodes, or combinations of these. Methods for constructing static structures are well known in the literature as *packing* or *bulk loading* techniques.

The Packed R-Tree The first packing algorithm for R-trees was proposed by Roussopoulos and Leifker [20]. Let the data-set contain N rectangles and each page can store up to M rectangles. First, sort the rectangles in the data-set according to the x-coordinate of their centroid (equivalently, the x-coordinate of the lower-left corner can be used). Pack the rectangles of the data-set into $\lfloor N/M \rfloor$ consecutive groups of M rectangles (except the last group). Recursively pack the resulting groups into nodes. The recursion continues until a single root node is created.

The Hilbert Packed R-Tree The Hilbert Packed R-tree [12] resembles its dynamic counterpart, because it utilizes the concept of the Hilbert space-filling curve. The algorithm builds a static R-tree with nearly 100% storage utilization. The tree is constructed in a bottom-up manner. First, the rectangles are sorted using the Hilbert space filling curve of their centroids. Then, they are grouped recursively into nodes, like in the Packed R-tree.

The STR R-Tree STR (Sort-Tile-Recursive) is a bulk-loading algorithm for R-trees proposed by Leutenegger et al. [17]. Let N be a number of rectangles in two-dimensional space. The basic idea of the method is to tile the space by using VS vertical slices, so that each slice contains enough rectangles to create approximately $\sqrt{N/M}$,

where M is the R-tree node capacity. Initially, the number of leaf nodes is $n_l = \lceil N/M \rceil$. First, the rectangles are sorted with respect to the x-coordinate of the centroids and VS slices are created. Each slice contains $VS \cdot M$ rectangles, which are consecutive in the sort order. In each slice, the objects are sorted once more, this time by the y-coordinate of the centroids and are packed into nodes (placing M objects in a node). The procedure continues recursively.

Time-Evolving R-Trees

There are many applications that manage data whose geometry changes over time. Examples include global change data (climate and land cover changes), transportation (traffic surveillance and intelligent transportation systems), social data (demographic and health), and multimedia data (movies). In such applications, object positions and/or extents change over time. A straightforward approach for indexing such data is to consider time as another dimension and use traditional R-trees. Nevertheless, this approach leads to extensive object overlapping (due to the long time intervals that have to be indexed) and, hence, deteriorated query performance. Thus, special R-trees have been developed for time-evolving applications.

The Segment R-tree attempts to resolve this problem by fragmenting long intervals into smaller, more manageable ones [15]. Two other approaches proposed are the *Overlapping* and the *Multiversion* R-trees. Consider a collection of spatial objects at two consecutive time instants t_1 and t_2; let one R-tree indexing the objects at t_1 and a second R-tree at t_2. If there are few changes between instants t_1 and t_2, then the corresponding R-trees will not be very different. The Overlapping R-trees take advantage of this observation by building a single structure for all consecutive R-trees that share common sub-trees. Multiversion R-trees use the notion of *partial persistence*. A data structure is called *persistent* if an update creates a new version of the data structure while previous versions are retained and can still be traversed [4]. *Partial* persistence implies that only the newest version of the structure can be modified. In the partial persistence approach, the sequence of R-trees can be visualized as the temporal evolution of the initial R-tree (the sequence of updates to the initial structure). The major advantage of partial persistence is that it uses space linear to the number of changes in the evolution of the objects while it provides access to any time instant with the same asymptotic efficiency of a dedicated R-tree for that time instant. Multiversion R-trees were introduced by Kumar et al. [16] for bi-temporal objects. Kollios et al. [14] applied them for spatio-temporal objects. Optimized approaches appear in [10,24].

A related application is that of indexing moving objects. Objects move with a known (typically linear) function of time, updating their position and movement function characteristics when necessary. One method proposed for indexing moving objects is the Time-Parametrized R-tree (TPR-tree) which indexes the current and anticipated future positions of objects [21]. A characteristic of TPR-trees is that they employ time parametrized MBRs that evolve over time as the object they contain move.

Query Optimization

Determining the best execution plan for a spatial query requires tools for estimating the number of data items that are retrieved by a query as well as its I/O and CPU cost. For R-tree indices, cost-based optimization exploits analytical models and formulas that predict the number of hits among the entries of the R-tree (called *selectivity*) and the cost of a query retrieval, measured in node accesses (or actual disk accesses, assuming existence of a buffering scheme).

The expected height h of an R-tree is $h = \log_f \frac{N}{M}$, where f is the average fan-out for parent nodes, M is the capacity of leaf nodes, and N is the number of data entries. The average cost C_W of a range query with respect to a query window $q = (q_1, \ldots, q_d)$, (for a d-dimensional R-tree), and provided that the sides $(s_j, 1, \ldots, s_j, d)$ of each node s_j are known is:

$$C_W(q) = \sum_j \left\{ \prod_{i=1}^{d} (s_{j,i} + q_i) \right\}. \quad (1)$$

Equation 1 allows the query optimizer to estimate the cost of a range query (measured in number of node accesses), if the MBR of each node s_j of the R-tree can be measured. The equation intuitively presents the relation between the sizes of the R-tree nodes and of the query window with the cost of a range query. Moreover, the influence of the node perimeters is revealed, explaining the efficiency of the R*-tree [6].

In order to avoid having to compute the R-tree node extends for estimating the cost of a query, the fractal dimension of a data-set can be used. The fractal dimension is a simple way to describe non-uniform data-sets, using just a single number. According to the model proposed in [5], the estimation of the number of disk accesses at level 1 (i. e., leaf level), denoted by $C_W(1, q)$, is given by:

$$C_W(1, q) = \frac{N}{f} \cdot \prod_{i=1}^{d} (s_{1,i} + q_i) \quad (2)$$

where $s_{1,i} = (f/N)^{1/fd}$, $\forall i = 1, \ldots, d$ and f is the average fan-out of the R-tree nodes.

A different approach is to use another property of the data-set, called *density surface* [26]. The density D of a set of N rectangles with average extent $s = (s_1, \ldots, s_d)$ is the average number of rectangles that contain a given point in d-dimensional space. Equivalently, D can be expressed as the ratio of the global data area over the work space area. Consider a unit workspace $[0, 1)^d$ then the density $D(N,s)$ is given by the following formula:

$$Den(N, s) = \sum_{N} \prod_{i=1}^{d} s_i = N \cdot \prod_{i=1}^{d} s_i. \quad (3)$$

Given the observations that: 1. The expected number of node accesses $C_W(q)$ for a query q is equal to the expected number of intersected nodes at each level; 2. The average number of intersected nodes is equal to the density D of the node rectangles inflated by q_i at each dimension; 3. The average number of nodes n_j at level j is $n_j = N/f^j$, where N is the cardinality of the data-set and f is the fan-out; 4. The density D_j of node rectangles at each level j can be expressed as a function of the density of the data-set, the following formula can be derived:

$$C_W(q) = \sum_{j=1}^{1+\log_f \frac{N}{f}} \left\{ \frac{N}{f^j} \cdot \prod_{i=1}^{d} \left(\left(Den_j \cdot \frac{f^j}{N} \right)^{1/d} + q_i \right) \right\}. \quad (4)$$

To reach this formula, square node rectangles and uniform data distribution for both data and node rectangles are assumed. Further work has provided formulas that drop the uniformity assumption both for data and internal nodes.

Key Applications

Processing Spatial Queries

R-trees can be used for processing basic and complex spatial queries. Basic query types include range and topological queries (for identifying data that satisfy a query region, along with their topological relations with respect to this region), nearest-neighbor queries (for identifying data closest to a query point), and spatial join queries (for finding pairs of data that satisfy a condition, e. g., "determine all hotels near a given location"). Complex queries include categorical range queries (identifying groups of objects that intersect with the query), reverse and constrained nearest neighbor queries (identify the set of objects that have the query point as a nearest neighbor), multi-way spatial join queries (joining between multiple inputs), incremental distance-join (find a subset of the Cartesian product that satisfies an order based on distance), closest-pair queries (find the k pairs of objects with the smallest distance), and all-nearest neighbor queries (for every object in set A identify its nearest neighbor in set B). Efficient implementations of R-tree variants for the above queries can be found in [25].

Spatio-temporal Databases

Spatio-temporal database systems aim at combining the spatial and temporal characteristics of data. There are many applications that benefit from efficient processing of spatio-temporal queries such as: mobile communication systems, traffic control systems (e. g., air-traffic monitoring), geographical information systems (GIS), multimedia, and location-based services (LBS). The common basis of these applications is the requirement to handle both the space and time characteristics of the underlying data. These applications pose high requirements concerning the data and the operations that need to be supported, and therefore new techniques and tools are needed for increased processing efficiency. A large number of specialized indexes for spatio-temporal data management are based on R-trees. Efficient implementations of spatio-temporal R-tree variants are found in [9].

Multimedia Databases

Multimedia database management systems aim at the effective representation and the efficient retrieval of multimedia objects, such as text, images, audio, and video. The basic characteristics of multimedia objects that makes multimedia query processing challenging are the following. They are characterized by significant complexity due to their rich content, and hence are hard to be organized and managed. The notion of similarity between two multimedia objects is often difficult to express. Therefore, new algorithms are required to search multimedia databases based on content. R-trees have been successfully used to alleviate these problems.

Data Warehousing and Data Mining

Data warehouses are specialized databases that serve as repositories for multiple heterogeneous data sources, organized under a unified schema to facilitate decision making. On-Line Analytical Processing (OLAP) is an analysis technique performed in data warehouses, which is exploratory-driven. Data mining (or knowledge discovery in databases) is the process of extracting interesting information or patterns from data in large databases. Both data warehousing and data mining need to access large amounts of data. For this reason, the acceleration of data access is facilitated by indexes. Although specialized indexes have been developed for these new environments (e. g., bitmap

indexes), R-tree-like indexes have also been successfully used.

Future Directions

The R-tree is one of the most influential Spatial Access Methods and has been adopted as the index of choice in many research works regarding spatial and multidimensional query processing. Taking into consideration the work performed so far, the R-tree stands for the spatial databases what the B-tree represents for alphanumeric data types. Considering the rich work performed on R-trees so far, it clearly covers almost all aspects concerning a database system: query processing, query optimization, cost models, parallelism, concurrency control, and recovery. This is the main reason why gradually database vendors adopted the R-tree for spatial data management purposes. Conclusively, the R-tree family is of great theoretical interest from several points of view. However, in view of new emerging technologies and applications, the R-tree will become a ubiquitous data structure. Taking into consideration the significant research performed on R-trees during the last 20 years, one could conclude that there is no more room for improvements or variations. However, current research in the area proves exactly the opposite. There are new and exciting application domains that require efficient representation and processing of objects that can be handled by R-tree-based access methods. Modern areas of increasing importance, such as P2P systems, data streams, and bio-informatics, are making use of R-trees as a powerful data management mechanism. It is sound to anticipate that in the future R-trees will be granted an exalted position in modern systems and applications.

Cross References

- ▶ Continuous Queries in Spatio-temporal Databases
- ▶ Indexing and Mining Time Series Data
- ▶ Indexing, High Dimensional
- ▶ Indexing, Hilbert R-tree, Spatial Indexing, Multimedia Indexing
- ▶ Indexing Schemes for Multi-dimensional Moving Objects
- ▶ Indexing Spatio-temporal Archives
- ▶ Indexing the Positions of Continuously Moving Objects
- ▶ Indexing, X-Tree
- ▶ Mobile Object Indexing
- ▶ Nearest Neighbor Queries in Network Databases
- ▶ Nearest Neighbor Query
- ▶ Pyramid Technique
- ▶ Quadtree and Octree
- ▶ Queries in Spatio-temporal Databases, Time Parameterized
- ▶ R*-tree
- ▶ Spatial Data, Indexing Techniques
- ▶ Trip Planning Queries in Road Network Databases

Recommended Reading

1. Ang, C.-H., Tan, T.C.: New linear node splitting algorithm for r-trees. In: Proc. of Symposium on Advances in Spatial Databases (SSD), Berlin, Germany, 15–18 July 1997, pp. 339–349
2. Beckmann, N., Kriegel, H., Schneider, R., Seeger, B.: The R*-tree: An efficient and robust access method for points and rectangles. In: Proc. of ACM Management of Data (SIGMOD), New Jersey, USA, 23–25 May 1990, pp. 220–231
3. Brakatsoulas, S., Pfoser, D., Theodoridis, Y: Revisiting r-tree construction principles. In: Proc. of the East European Conference on Advances in Databases and Information Systems, Bratislava, Slovakia, 8–11 Sept 2002, pp. 149–162
4. Driscoll, J.R., Sarnak, N., Sleator, D.D., Tarjan, R.E.: Making data structures persistent. J. Comp. Syst. Sci. **38**(1):86–124 (1989)
5. Faloutsos, C., Kamel, I.: Beyond uniformity and independence: analysis of r-trees using the concept of fractal dimension. In: Proc. of ACM Symposium on Principles of Database Systems (PODS), Minnesota, USA, 24–26 May 1994, pp. 4–13
6. Faloutsos, C., Sellis, T., Roussopoulos, N.: Analysis of object oriented spatial access methods. SIGMOD Record **16**(3):426–439 (1987)
7. Garcia, Y.J., Lopez, M.A., Leutenegger, S.T.: On optimal node splitting for r-trees. In: Proc. of Very Large Data Bases (VLDB), New York, USA, 24–27 Aug 1998, pp. 334–344
8. Guttman, A.: R-trees: A dynamic index structure for spatial searching. In: Proc. of ACM Management of Data (SIGMOD), Massachusetts, USA, 18–21 June 1984, pp. 47–57
9. Hadjieleftheriou, M., Hoel, E., Tsotras, V.J.: Sail: A library for efficient application integration of spatial indices. In: Proc. of Scientific and Statistical Database Management (SSDBM), Santorini Island, Greece, 21–23 2004, pp. 135–138
10. Hadjieleftheriou, M., Kollios, G., Tsotras, V.J., Gunopulos, D.: Efficient indexing of spatiotemporal objects. In: Proc. of Extending Database Technology (EDBT), Prague, 24–28 Mar 2002, pp. 251–268
11. Huang, P.W., Lin, P.L., Lin, H.Y.: Optimizing storage utilization in r-tree dynamic index structure for spatial databases. J. Syst. Softw. **55**(3):291–299 (2001)
12. Kamel, I., Faloutsos, C.: On packing r-trees. In: Proc. of Conference on Information and Knowledge Management (CIKM), Washington DC, USA, 1–5 Nov 1993, pp. 490–499
13. Kamel, I., Faloutsos, C.: Hilbert r-tree: An improved r-tree using fractals. In: Proc. of Very Large Data Bases (VLDB), Santiago de Chile, Chile, 12–15 Sept 1994, pp. 500–509
14. Kollios, G., Tsotras, V.J., Gunopulos, D., Delis, A., Hadjieleftheriou, M.: Indexing animated objects using spatiotemporal access methods. IEEE Transactions Knowl. Data Eng. (TKDE) **13**(5):758–777 (2001)
15. Kolovson, C., Stonebraker, M.: Segment Indexes: Dynamic indexing techniques for multi-dimensional interval data. In: Proc. of ACM Management of Data (SIGMOD),Colorado, USA, 29–31 May 1991, pp. 138–147
16. Kumar, A., Tsotras, V.J., Faloutsos, C.: Designing access methods for bitemporal databases. IEEE Transactions Knowl. Data Eng. (TKDE) **10**(1):1–20 (1998)

17. Leutenegger, S.T., Edgington, J.M., Lopez, M.A.: Str: A simple and efficient algorithm for r-tree packing. In: Proc. of International Conference on Data Engineering (ICDE), Birmingham, UK, 7–11 Apr 1997, pp. 497–506
18. Manolopoulos, Y., Nanopoulos, A., Papadopoulos, A.N., Theodoridis, Y.: Rtrees: Theory and Applications. Springer, Germany (2005)
19. Nanopoulos, A., Vassilakopoulos, M., Manolopoulos, Y.: Performance evaluation of lazy deletion methods in r-trees. GeoInformatica **7**(4):337–354 (2003)
20. Roussopoulos, N., Leifker, D.: Direct spatial search on pictorial databases using packed r-trees. SIGMOD Record **14**(4):17–31 (1985)
21. Saltenis, S., Jensen, C.S., Leutenegger, S.T., Lopez, M.A.: Indexing the positions of continuously moving objects. SIGMOD Record **29**(2):331–342 (2000)
22. Schreck, T., Chen, Z.: Branch grafting method for r-tree implementation. J. Syst. Softw. **53**(1):83–93 (2000)
23. Sellis, T., Roussopoulos, N., Faloutsos, C: The r+-tree: A dynamic index for multi-dimensional objects. In: Proc. of Very Large Data Bases (VLDB), pp. 507–518 (1987)
24. Tao, Y., Papadias, D.: MV3R-Tree: A spatio-temporal access method for timestamp and interval queries. In: Proc. of Very Large Data Bases (VLDB), pp. 431–440 (2001)
25. Theodoridis, Y.: The R-tree-Portal. (2003)
26. Theodoridis, Y., Sellis, T.: A model for the prediction of r-tree performance. In: Proc. of ACM Symposium on Principles of Database Systems (PODS), pp. 161–171 (1996)

Rubber Sheeting

► Positional Accuracy Improvement (PAI)

Rubber-Sheeting

► Conflation of Features

Rules-Based Processing in GIS

MICHAEL SANDERSON, PETER WOODSFORD
Laser-scan, Cambridge, UK

Synonyms

Digital mapping

Definition

This paper describes how Laser-Scan developed products to meet the GIS market requirements during the period 1980–2007. From an early background in the intelligent vectorization of maps, Laser-Scan's product philosophy has been to provide solutions that maximize the degree of automation in spatial data handling. The ever-increasing size of spatial databases mandates this approach, as well as the risk of human error associated with manual intervention. However, it is fully recognized that human interaction is required to handle cases that cannot be resolved. Much of the automation proceeds from the application of algorithms. Topology management plays a significant role. Successive generations of product have been characterized by increasing reliance on standards and mainstream database technology and by more formalized implementations of rules-based processing. Typical applications include data quality assessment, validation and maintenance services, multi-product information management and generalization. Up to date information is available via [5].

Historical Background

Laser-Scan's GIS products have evolved across three generations, characterized by increasing alignment with mainstream Information Technology and hence by the exploitation of the dramatic advances in hardware, software and IT infrastructure that have taken place since the company was formed in 1969.

The first generation products were essentially digital mapping systems, and were based on the company's capabilities in automated data capture from maps. Initially this used special purpose laser-scanning technology, to follow lines and edges and to recognize symbols on film versions of maps. Later the algorithms were adapted to extract intelligent, quality assured vector information from raster data [4] in the VTRAK product. VTRAK was complemented by a powerful vector editing system that formed LAMPS (Laser-Scan Automated Map Production System). Vector data were held in files, in a proprietary format with a publicly accessible API. Raster edit and burn-in capabilities were also supported. LAMPS was integrated with powerful map finishing systems such as Mercator (now part of STAR INFORMATIC). A number of cartographic and charting systems remain in production use with considerably more than 20 years of service. Scripts and macros were used extensively to control sequences of automated operations on the data and to enforce business rules, but these were in not formalized in any meaningful sense.

The advent of the object-oriented (O-O) paradigm provided a more formalized platform for rules-based processing and handling of structured information. It embodied the thinking behind the mantra:

$$\mathbf{M}\text{odel} = \mathbf{D}\text{ata} + \mathbf{S}\text{tructure} + \mathbf{A}\text{lgorithm}\ [13]$$

Starting in the late 1980s Laser-Scan developed the Gothic geospatial object-oriented processing environment. This

approach was similar to Intergraph's TIGRIS and to Smallworld GIS.

Gothic represents real world objects by object models in classes, according to a schema. Instances of objects possess identity, geometry, attributes and behaviors. In addition relationships between objects, including topology, can be represented and complex objects (objects made up of a group of objects) constructed and maintained. A comprehensive set of geoprocessing operations is provided, together with support for object lifecycles. A proprietary user language (LULL) was developed as both the scripting language and the methods language in the Gothic object database. The use of methods led to the formalization of business logic, constraints and quality rules and their enforcement centrally across all applications using the rules-processing environment provided by the Gothic database.

Two concepts of the object lifecycle approach are worthy of further discussion, since they point the way to the future architecture. They are encapsulation and polymorphism. Encapsulation standardizes and simplifies access to data objects and protects them. The data stored inside the objects can be manipulated safely by recourse to the methods, which the object class provides or inherits. For query, complex derived attributes (e. g. area) may be computed from object data (e. g. geometry), simplifying maintenance. Typically the method behavior is of sufficient complexity that attempting to reproduce it in application logic risks data corruption or performance degradation. Most importantly, encapsulation promotes thin clients. Polymorphism is the ability to use the same syntax for objects of different types. A good example is a validation method. When editing data, the application may wish to validate the feature being edited before committing it to the database. Depending upon whether the feature is of type motorway (inherits the road validation method) or of type contour line (inherits the relief validation method) it will invoke a completely different behavior. Provided the validate method is polymorphic the application need know nothing about the internals of how validation is accomplished. Again thin client delivery is appropriate because all the geoprocessing has been executed in the database and can be shared and accessed by all clients. Polymorphism and encapsulation are well suited therefore to the current web based IT architecture.

Two aspects of Gothic have become less viable. The use of a proprietary language was not sustainable in the face of the widespread adoption of O-O programming languages such as Java. LULL has given way to Java, firstly as the scripting language and more recently as the methods language. Thus a modern Gothic application such as the Air Information Management System is written at the application level entirely in Java. Equally the use of a proprietary database became unacceptable as users sought the integration of their geospatial information holdings with all other business information. Enterprise databases are delivered with features such as security and scalability as standard and with a large pool of supporting specialists. Rather than replicating these features, it is better to seek the GIS value add in managing the spatial data.

The database storage issue has resulted in a new Laser-Scan architecture. The re-architecture is based on a mainstream database (typically Oracle, with the advent of Oracle Spatial). However it is something more fundamental than simply that. It is more fully understood as a requirement for the deployment of powerful geo-processing capabilities as components in an open, standards-based architecture. This provides the rationale for the Radius suite products, launched in 2002.

Scientific Fundamentals

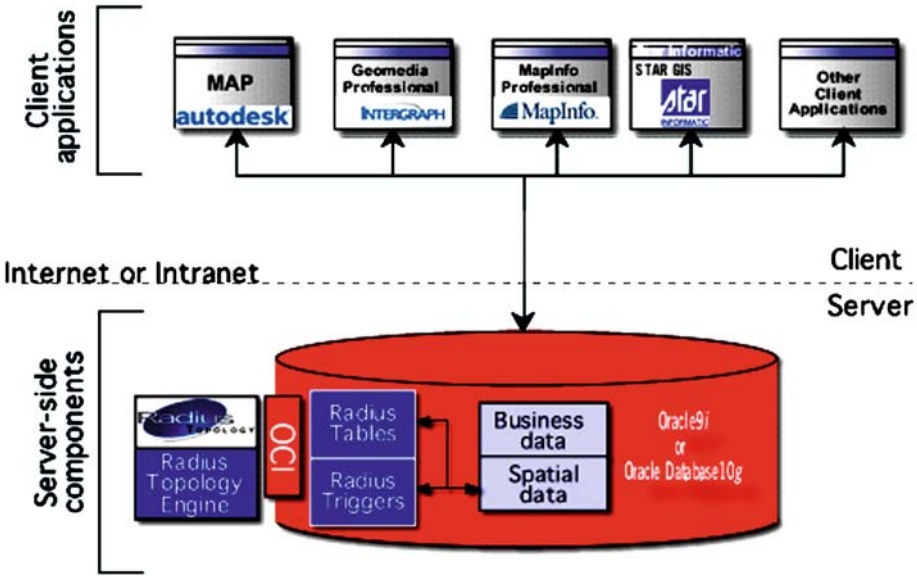
The O-O techniques combined with persistent topology provide the basis for managing spatial data. This section opens with a description of key functionality in the Radius product suite. As described above the Radius products are a subset of the Gothic environment.

Radius Topology

The Laser-Scan topology engine implements the 2D components of the ISO19107 topology model [3]. Radius Topology delivers powerful data cleaning, connectivity and analysis functionality to spatial data held in Oracle9*i* or Oracle Database10*g*, through the SDO_Geometry interface [12]. The design (see Fig. 1) is based on avoiding any impact on other GIS applications reading/writing the standard spatial data types in Oracle according to whatever long transaction model has been determined. There is a very fine level of control over topological relationships, which can be defined and prioritized on an individual class-class basis, and which are maintained 'on-the-fly' across all processes. In addition, support is provided for directly editing topological primitives.

In the real world GIS data has been collected with a height value (sometimes known as 2.5D data). Radius Topology therefore implements topological structuring with a height (or z value) for each vertex, using vertical snapping tolerances and maintaining relationships such as 'over/under'.

Over time support for specific vertical market applications has emerged. Thus pre-configured tolerances and priorities have been defined to support utility connectivity, network conflation, land and property gazetteer creation and maintenance. These are termed Wrappers and provide a simple Java user interface and a PL/SQL API.



Rules-Based Processing in GIS, Figure 1 The Radius Topology Architecture

Radius Studio

Topological structuring is an automated process. Often what is required is a meaningful assessment or statement of the data quality position – a window on the data. This assessment can be used to define remedial programmes and to assess the fitness for purpose of the data set in particular decision-making scenarios. Radius Studio is such an assessment tool. It makes use of the 3-tier architecture that is emerging as part of the web services (component) paradigm. This means it has to operate within a standards-based framework. It is based on the following de facto and de jure standards:

- Component model – J2EE/EJB;
- Service model – ISO 19119;
- Geometry model – ISO 19107;
- Domain modelling – W3C OWL;
- Rules language – W3C SWRL.

Radius Studio consists of a rules browser/editor, conformance checker, a spatial data processing and reconciliation tool and a task sequencing tool. It has been designed on knowledge management principles [8] to allow the domain expert to define the business rules. It offers an approach to semantic interoperability, as well as measurable spatial data quality control. A common language interface is used in preference to a programmatic interface (see Fig. 2).

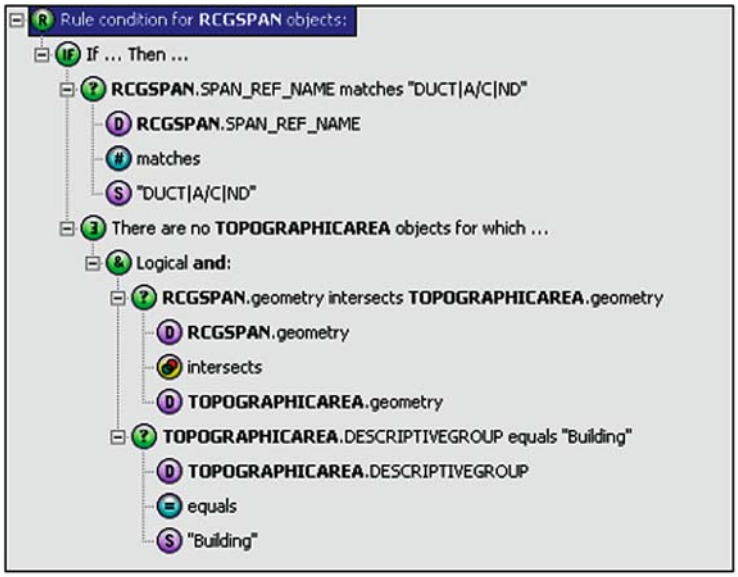
Gothic

Laser-Scan offers a range of technologies based on its Gothic O-O database, from an application development environment to tools for web mapping. The Gothic Database Server plays a central role, providing access management for a versioned, object-oriented spatial database (OODB). Spatial objects, in 2D or 3D, are members of object classes defined in a schema. Methods are defined for object classes – these are rules or behaviors written in the internal language or in Java. Methods have access to the full range of geospatial processing capabilities. Encapsulation is supported within the object-lifecycle framework. Combining data and functionality in this way ensures that Gothic datasets are self-validating. Rich data models with complex relationships, rules and constraints can be implemented and maintained across a wide set of applications.

Gothic databases are seamless and can hold very large continuous datasets with many millions of objects. The topology engine is the same set of libraries that are described above for Radius Topology only stored and accessed in a different manner. Objects can be stored in geographical coordinates, with whole earth support (crossing polar regions and dateline), or in any coordinate projection. They can be stored in one coordinate system, structured and queried in another, and displayed in another. A full set of projection and map transformations are supported. States of an object can be date-stamped to support temporal evolution. In addition to vector objects, Gothic databases can hold and manage raster and image objects.

The Gothic Developer Application Programming Interface (API) provides a bridge between Laser-Scan's O-O spatial database capabilities and the requirements of application builders. Using Developer, applications that access the Gothic O-O database can be written or customized.

LAMPS2 is an advanced production application for spatial data sets, maps, charts, plans or any other geographical data product. It separates the tasks of compilation and update from those of presentation and product generation,



Rules-Based Processing in GIS, Figure 2 A fragment of a rule definition in Radius Studio

and supports data re-engineering and structure-building. Raster/vector integration includes raster edit and burn-in capabilities.

Gothic JADE offers a completely extensible Java-based desktop application providing spatial and mapping capabilities. It provides access to the Gothic spatial toolkit and object database from a framework application into which standard or user-written modules can be loaded.

Key Applications

A subset of applications is described, with an emphasis on the role of rules-based processing and starting with the Radius suite.

Topology and its Uses

Topology is defined elsewhere. Using indexing Laser-Scan utilizes the SDO Geometry construct in Oracle to calculate and persist topology on the server side. The main reason for this is to provide interoperability with business applications. An alternative view is described in [2]. The uses of topology in GIS include the analysis and maintenance of networks, routing applications, efficient maintenance of shared boundaries in for example land parcel applications and the speeding up of spatial database queries [15].

Positional Accuracy Improvement (PAI)

Positional accuracy is one of the data quality elements covered by ISO 19113. The advent and widespread use of high precision GPS has meant that positional inaccuracies in historical survey data have been exposed and become a problem. The vast majority of the spatial data in use were collected before GPS was ubiquitous. In Europe alone the value of these data were €36bn in 1999 [1]. Topology can be used effectively to solve part of this problem by establishing the relationships in the data before applying shift vector transformations to improve positional accuracy. This allows the pre-GPS data to be re-used. A Radius Topology PAI Wrapper tracks and maintains the relationships between asset data and base data, keeps asset data connected during shifting and audits relationships before and after shift.

Data Quality Certification and Maintenance

There is a fundamental issue over fitness for purpose in the use of spatial data in the decision-making process. In every day use or in natural disaster response management the value of the data are undermined if not updated and managed over time. In a similar manner to the GPS issue discussed above, previous generations of software tools have seen the business rules of an organization locked into application code, in graphical attribution. In addition there has been little rigour in spatial data administration. To create true knowledge economies business rules need to be abstracted and made available throughout the enterprise by the domain experts. Most organizations are asset intensive. Assets have spatial attributes. These spatial data need to conform to certain business rules. Also

the cost of collection and subsequent maintenance of these data needs to be measured against quality measures. Examples of such measures may be the number of dry holes that are dug to locate the company's plant, or the number of customer refunds paid as a result of outages not identified in network maintenance or repair programmes. A rules-based approach in the latter example (see the rule in Fig. 2) would establish the connectivity between the property and the asset as a data management problem. The results of these analysis will in the future need to be a part of the key performance indicators against which an organization measures itself.

Multi-product Information Management Systems

Concentration of business logic on the server side, together with rich data modelling capabilities, underpins the implementation of multi-product information management systems. The key characteristics are once only database maintenance with assured data quality, multi-product capabilities and consistent and efficient product provision. An example is the Aeronautical Production System (APS) [6]. This produces a quality assured set of Flight Information products including cartographic products (en route charts), data products (DAFIF, DVOF) and textual reports automatically from a single database.

In APS a formally defined data model (DAFIF) drives both the data model schema and the lifecycle and validation rules (see Fig. 3). All data entering the system is validated against the DAFIF rules. Product Data Rules extract information from the database by combinations of operations such as selection of a temporal snapshot of a localized geographic region, transformation of specific features and attributes and topological structuring of geometric features. Product Cartographic Rules define automatic and intelligent feature portrayal. An example is shown in Fig. 4.

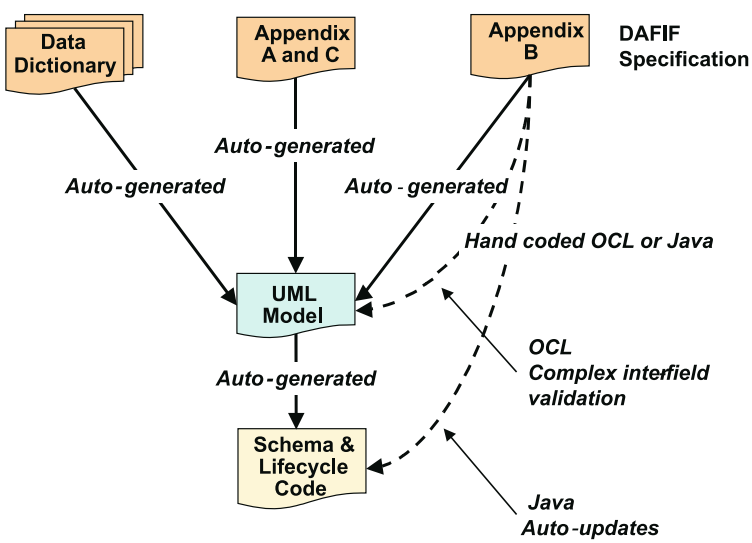
Data Model Re-engineering

Data providers sometimes take a quantum leap and seek to re-purpose their data by adopting a richer and more capable model. A move from spaghetti data to structured real world objects with persistent identifiers is a prime example of such re-purposing. Rules-based processing is key to achieving such re-engineering in a manageable timescale and with assured quality. A major example is the Ordnance Survey Great Britain (OSGB) project to re-engineer Land-Line data (220,000+ sheets of points and lines) to MasterMap® data (continuous, 0.5 billion richly attributed polygon objects with public identifiers (TOIDs™)). This was achieved by an automated flowline using Gothic in one year, with less than 1% manual editing [11].

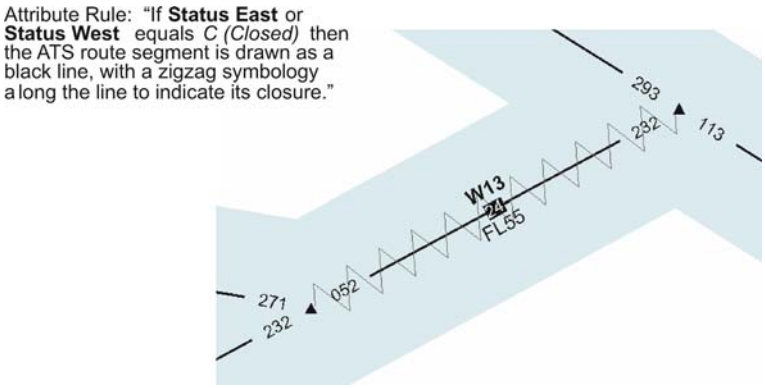
Generalization

Laser-Scan's *Clarity* is a cartographic and data generalization product built on the Gothic intelligent geo-spatial functionality. It automatically reduces data resolution (and hence volume) whilst retaining essential information content by applying generalization rules [9]. For cartographic purposes Clarity can also apply rules to enhance graphical clarity and readability using active, goal-seeking, 'self aware' agents. This can result in the displacement of features, in addition to other generalization processes such as selection and amalgamation.

Processing rules and constraints that must be maintained are defined by a simple user interface and encoded in



Rules-Based Processing in GIS, Figure 3
Aeronautical Production System – model driven definition of schema and rules base



Rules-Based Processing in GIS, Figure 4 A cartographic rule on an aeronautical chart detail

XML. Generalization is performed automatically, but with exception handling. Instances that cannot be processed automatically or which do not conform to product rules are flagged for subsequent operator action.

Data generalization for example from 1:10K to 1:50K in the German Länder project [14] is typically achieved with complete automation. Cartographic generalization still involves some operator action but time savings in a production environment of an order of magnitude have been achieved, e. g. IGN France Carto2001 and New Base Map projects [7].

Future Directions

Much work has been carried out on spatial data models, by the OGC and others, but there are several areas where future progress is essential. There are five areas that are of significance: Progress towards 3D capabilities; Master Data Management; spatial as mainstream data, ontologies and semantics and Service Oriented Architecture (SOA).

3D

Progress towards 3D capabilities is rapid, spurred on by the advent of Google Earth. From our perspective the key issues are the extension of the topology engine, which currently handles 3D data but restricted to 2.5D topology, to a full 3D capability, in line with ISO19107.

Master Data Management

There is little real progress towards interoperability. Significant work has been undertaken in mainstream supply chain processes to establish the concept of Master Data. Central control is required to establish meta referencing to ensure that all participants in the supply chain have a common understanding of transactional events. It is essential to have high quality base reference mapping to provide the gold standard for data and against which to align any incoming or third party data.

Integration to Business Intelligence (BI)

Further work on the smooth integration of spatial models with other domains, particularly BI is essential if the spatial community is to be able to demonstrate its capability in delivering benefits across the organization. There is no reason why C-level executives current predilection for managing performance through dashboards, should not include spatial data measures. Given the expenditure on spatial data [1] there must be measurement. A Topological Quality Assessment Service is being implemented in the OGC OWS-4 testbed as an initial step to defining interface standards for quality assessment services [10].

Ontologies and Semantics

The model driven approach will become more powerful and widespread as better tools for managing ontologies emerge. Radius Studio can currently use an ontology defined in OWL but few of these are yet available. These developments will also result in richer discovery metadata and progress towards the Semantic Web in the geospatial domain.

Web Services Architecture

Radius Studio can be used over the web, as a service. The approach by Laser-Scan to the component-based web architecture has led towards a different business model for the company. This new approach is based on partnering. Some 37 years after Laser-Scan was founded this new business model has led to the Radius brand assuming greater importance than the company name, which was changed at the end of 2006 to 1Spatial.

Cross References

► Computing Fitness of Use of Geospatial Datasets
► Conflation of Geospatial Data

- ▶ Geospatial Semantic Integration
- ▶ Geospatial Semantic Web, Interoperability
- ▶ Map Generalization
- ▶ OGC's Open Standards for Geospatial Interoperability
- ▶ Ontology-Based Geospatial Data Integration
- ▶ Oracle Spatial, Geometries
- ▶ Positional Accuracy Improvement (PAI)

Recommended Reading

1. Commercial Exploitation of Europe's Public Sector Information. PIRA International, Leatherhead, UK (2000)
2. Hoel, E., Menon, S., Moorhouse, S.: Building a Robust Implementation of Topology. In Advances in Spatial and Temporal Databases. Proceedings of the 8th International Symposium on Spatial and Temporal Databases. SSTD 2003. Santorini, Greece (2003)
3. ISO 19107:2003. Geographic Information – Spatial schema. Available via http://www.isotc211.org/
4. Jackson, M.J., Woodsford, P.A.: GIS data capture hardware and software. In: Geographical Information Systems, pp. 239–249. Longman Scientific and Technical, Harlow, England and Wiley, New York (1991)
5. Laser-Scan. www.laser-scan.com and from December 2006, www.1spatial.com
6. Laser-Scan. The Aeronautical Production System (APS) (2003). http://www.laser-scan.com/applications/aps.htm
6. Lecordix, F., Trévisan, J., Le Gallic J.M., Gondol, L.: Clarity Experimentations for cartographic generalisation in production. 9th ICA Workshop of the ICA Commission on Generalisation and Multiple Representation, 25 June 2006 – Portland, USA. Available at http://ica.ign.fr/
7. Loshin: Enterprise Knowledge Management (The Data Quality Approach). Morgan Kaufmann, San Francisco (2001). ISBN 0-12-455840-2
8. Neuffer, D., Hopewell, T., Woodsford, P.: Integration of Agent-based Generalisation with Mainstream Technologies and other System Components. ICA Workshop on Generalisation and Multiple Representation 20–21 August 2004 Leicester, UK Also available at http://www.laser-scan.com/papers/Neuffer-v2-ICAWorkshop.pdf
9. OGC, Open Geospatial Consortium, Open Web Services, Phase 4 (2006), Topology Quality Assessment Service in Geoprocessing Workflow thread at http://www.opengeospatial.org/projects/initiatives/ows-4
10. Ordnance Survey Great Britain (GB). Meeting the Master Challenge. 18 Sept 2001. Available at: http://www.ordnancesurvey.co.uk/oswebsite/media/news/2001/sept/masterchallenge.html
11. Ravikanth, K.V., Godfrind, A., Beinat, E.: Pro Oracle Spatial: From Professional to Expert. Apress, Berkley (2005). ISBN 1590593839
12. Ross, D.T., Rodriguez, J.E.: Theoretical foundations for the computer aided design system, pp. 305–322. Proceedings of AFIPS 23, SJCC, Detroit (1963)
13. Urbanke, S.: AdV – Projekt "ATKIS – Generalisierung" 53. Deutscher Kartographentag, Rostock, 22 Sep 2005 (in German)
14. Watson P.: Topology and ORDBMS Technology, a White Paper, 2002. Available at http://www.laser-scan.com/pdf/topo_ordbms.pdf

Rural Hydrologic Decision Support

DAMON GRABOW
Regional Weather Information Center,
University of North Dakota, Grand Forks, ND, USA

Synonyms

Decision support; Rural hydrology; Soil and water assessment tool "SWAT"

Definition

Rural hydrologic decision support, in a general sense, is a tool or data set that supports decision makers with an emphasis on hydrology and its effects on other environmental factors. A hydrologic model such as the Soil and Water Assessment Tool (SWAT) could be referred to as a rural hydrologic decision support tool.

Historical Background

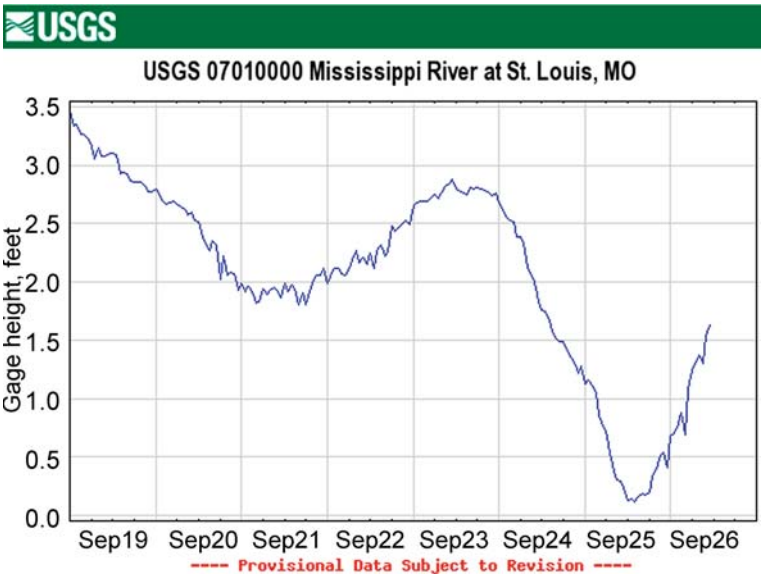
The typical data used for hydrologic decision support were streamflow datasets or possibly groundwater depth. For example, the United States Geological Survey (USGS) maintains a large number of stream gauges around the US. Figure 1 shows an example of stream gauge data output. Gauges, such as the one on the Mississippi River at St. Louis, MO, are placed along major river systems.

Scientific Fundamentals

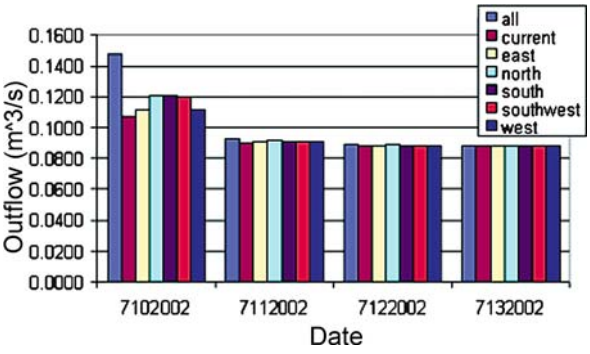
Rural Hydrologic Decision Support Example

One example of a rural hydrologic decision support is the growth of a small rural community during a climatic trend of heavy rainfalls. The community is looking to expand in the area with the least environmental impact. The example site is a rural community of approximately 1000 people that covers a land area of 100 hectares. The community is surrounded by farm fields allowing for ample growth provided the sale of the land is approved. There is a small coulee that runs through town with a late summer discharge of 0.1 $m^3 s^{-1}$. The only impervious surfaces are the rooftops and one paved road, all of the other roads and parking lots are gravel. Given the lack of impervious surfaces and size of the community is classified as low-density housing even though the lots are in 0.101 to 0.405 hectares in size.

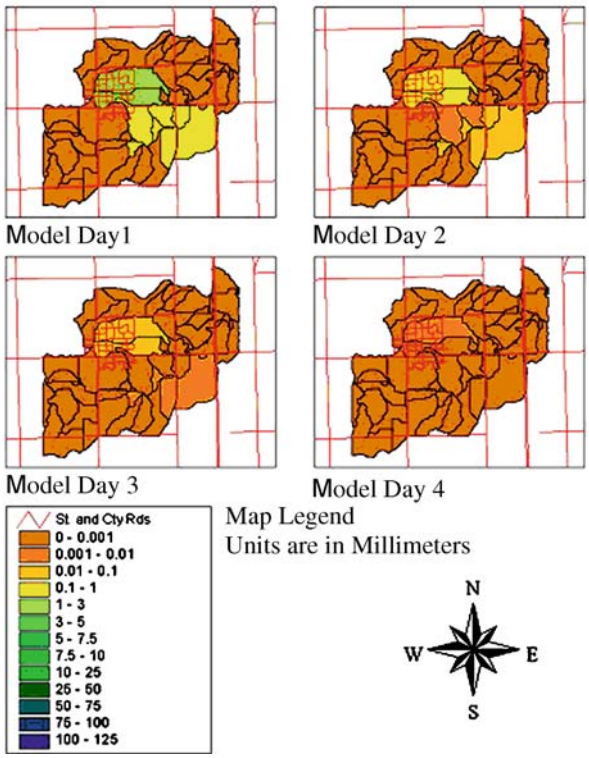
Topographically, the area is flat. A west to east slope is apparent as the western edge peaks at 267 m while the lower eastern edge shows an elevation of 259 m but has an average slope of 1.0–1.5 m per km. Due to the low slope,



Rural Hydrologic Decision Support, Figure 1 An example of a stream gauge data set. Data is from the USGS, http://waterdata.usgs.gov/mo/nwis/, gauge 70100000 along the Mississippi River at St. Louis, Missouri



Rural Hydrologic Decision Support, Figure 2 Total stream outflow from each urban scenario for 19.05 mm precipitation event



Rural Hydrologic Decision Support, Figure 3 Shown are spatial surface runoff results using the 19.05 mm precipitation event with the current urban area overlaid with current state and county roads

a few non-natural features are apparent in the digital elevation model; Railroad tracks and roadbeds are distinctly visible along with the interstate overpass.

For this example, 35 separate scenarios were established, 7 land use plans and 5 precipitations events. Along with the original land use, six possible urban land use areas were created for this study; south, north, east, southwest, west, and a combination of all of these. The precipitation events were of sizes 2.5 mm, 6.6 mm, 19.05 mm, and 44.45 mm, and 207 mm.

The results from this example experiment demonstrate the use a rural hydrologic decision support tool. Total stream outflow (Fig. 2) and estimated potential surface runoff (Fig. 3) were used to demonstrate the varying hydrologic impacts of land use change around the community.

By using the Soil and Water Assessment Tool (SWAT), one example of a hydrologic decision support tool, the community decision makers are able to visualize the areas of concern based on a wide variety of estimated variables. However, decision tools like SWAT need a compilation of large environmental data sets that include weather, hydrologic data, soils, land use and land cover, and elevation data.

R

A large chunk of these data can be obtained over the Internet for free depending on the users scaling needs.

Data

There are many pieces of data to gather in utilizing a decision support tool's full potential. For this example the following set of environmental data was gathered to initiate the support tool.

Growth Plan

If the user is a rural community it is highly beneficial to build a series of growth plans for comparison within the rural hydrologic decision support tool. Using a rural hydrologic decision support tool can estimate the environmental impacts of any land use changes or land use practices.

Meteorological Data

Meteorological data has a variety of variables. When dealing with hydrology precipitation is most common. Rain gauge data is the most widely used precipitation data type. Estimated precipitation from radar data is becoming available for the use within a GIS and provides for higher accuracy in hydrologic tools. However, other meteorological variables such as wind speed, temperature, cloud cover, dew point, and relative humidity can have an effect on the hydrologic changes over the course of time.

Hydrologic Data

Estimates of the existing surface water characteristics can be used to initialize a environmental model or equations. These estimates can consist of a stream flow rate, pond measurements, and even ground water levels. These data sets can be found on the web or estimated in the field with the use of some simple tools.

Soils

The types of soil can make a difference in the hydrologic response. Each soil type has its own hydrologic characteristics such as water capacity or infiltration rate, which are dependant upon the amount of clay, silt, or sand in the soil.

Land Use / Land Cover

The land use/land cover has an impact on the surface runoff, water quality, soil moisture, and even ground water accumulation. The land use can change the amount of impervious surface and have a direct impact to the hydrologic response. Vegetation types and patterns can alter the hydrologic response of a watershed as well. Any changes in land use and land cover should be addressed in the decision support data.

Elevation

Elevation data, typically found in grid form, is used to delineate watersheds and their sub-basins, locating outlets, and the hydrologic routing and flow patterns within the watersheds. It influences the speed at which the surface water will travel downstream and the erosion tendencies of the soil.

Key Applications

Rural hydrologic decision support can be used by any type of decision maker needing to make a decision relevant to hydrology.

Regional Planning/Rural Government

A large number of decision makers hold government offices and a large number of those offices represent rural areas. A hydrologic decision support tool/data set can be a benefit to a rural area and its growth and development plan. For example, by using the growth plan in a hydrologic decision support tool areas within that plan that produce the most hydrologic impact may be avoided. This may include topics such as land use management or flood retardants. Cau et al. (2003) used SWAT to estimate the water budget for the Sardinia region in Italy. Data for the project were a coarse soils map (1:250,000) and the CORINE Land Cover map (1:100,000), a 44-class vegetation and land use classification map. Regional rain gauge data and stream flow data were available dating back to 1922. The goal of the project was to construct a decision support tool that enables decision makers to use the water cycle as a variable in land use planning. Rainfall, evapotranspiration, and flow rate were the focus variables for the tool. By using the SWAT system for planning water management policies at a regional scale, decision makers can decide when, where and how water will be distributed. Cau et al. (2003) found that the model can play a leading role in estimating the water budget when the human utilization needs to be considered. A good description of the need for a rural hydrologic decision support system comes from a 2004 article written by Schär et al. This article explains the demise of the Aral Sea in central Asia that started in the early 1960's. Due to the agricultural demands in the Aral Sea watershed the rivers that replenished the sea from its evaporation were not carrying enough water to maintain the natural water levels. High water-demanding crops like cotton were the basis of the agricultural economy in the region. However, International governing bodies were able

to change the agricultural focus to less water-demanding crops such as fruits, vegetables, and livestock reducing the use of irrigation. In order to maintain the water levels in the region meteorological models and hydrologic decision support tools were implemented to curb water usage and better manage the resources.

Hydrology

Melesse et al. (2003) researched the effects of land use and land cover change on runoff using the United States Department of Agriculture, Natural Resources Conservation Service Curve Number (USDA-NRCS-CN). The analysis was performed on four individual years, 1984, 1990, 1995, and 2000. Land cover and land use data were taken from the Landsat Thematic Mapper and Enhanced Thematic Mapper Plus of the Etonia, Econlockhatchee, and S-65A sub-basins in Florida. Elevation data were a 30-meter pixel grid obtained from the USGS. The chosen storm events, recorded by rain gauges, produced volumes of no less than 12.75 mm. In comparing the alterations in land-cover over the study years, the predicted runoff depth was shown to increase. The increased impervious surface area in 1995 and 2000 increased the peak flow, and reduces the time to peak and time to recession, compared to 1984 and 1990, respectively (Melesse et al. 2003).

Chemistry

Water quality is an important aspect of rural hydrologic decision support. With the use of GIS and spatial data models water quality can be modeled along with changes in land use or cover, surface water runoff chemistry, or even precipitation and snow melt. Rural hydrologic decision support can aid in water quality estimates and calculations. Chaplot et al. (2004) used a hydrologic decision support tool to investigate nitrogen applications and variations in agriculture practices. The nitrogen was varied by percentages of +20, +40, –20, –40, and –60 and the land use scenarios varied from pastures, corn-soybean rotations, and continuous winter wheat practices. Chaplot et al. found quantitative predictions of the impact of farming practices on annual and seasonal flow, NO_3-N, and sediment discharges and also explain that this technology should assist scientists, policy makers, and farmers in making decisions for management of watersheds.

Geology

In the same sense of a support tool or data set for surface hydrology, a support tool for hydrogeology can be used. Spatially based ground water models help supply information to decision makers regarding the effects on ground water, this being subsurface flow or water quality. Gutzler and Nims (2005) give an excellent example of one use of a hydrologic decision support tool. They discuss the groundwater depletion in Albuquerque, NM. Although Gutzler and Nims don't apply a hydrologic decision support tool they describe a groundwater issue that decision makers need more information about and is an opportunity to utilize a rural decision support tool.

Sun and Cornish (2005) researched the Liverpool Plains in Australia. Using a rural hydrologic decision support tool Sun and Cornish found that the groundwater recharge might not be as high as originally found in previous studies. The spatial characteristics over the entire catchment were included within the study (soil types, land cover, soil moistures, etc.). Sun and Cornish found that not only land clearing and crop practices contributed to the groundwater recharge but that climate variances did as well.

Eckhardt et al. (2003), also using SWAT, researched climate change impacts on groundwater recharge in central Europe. This research looked at the increasing amount of CO_2 in relationship with plant physiology and its affect on the groundwater recharge and change in streamflow. However, Eckhardt et al. also explains the complexity of using a hydrological decision support tool in that slight changes in the atmospheric boundary conditions can have a significant impact on the findings.

Ecology

Plants and animals are affected by decisions made by humans. Thus a rural hydrologic decision support tool can provide an estimate impact to change in habitat or conduct vulnerability assessments. Whittaker (2005) compared two remediation policies for salmon habitat within the Columbia River basin in Washington and Oregon. The remediation policies were installed to reduce the amount of nitrogen in the river system by local agriculture. The first policy controlled the amount of nitrogen used by the farmer while the second policy taxed the farmer for the amount of nitrogen runoff produced. This research demonstrates the ability of a rural hydrologic decision support tool to aid in rural policies such as agricultural nitrogen usage for the benefit of flora and fauna.

Future Directions

There are many drawbacks for a rural hydrologic decision support system including, cost, technological expertise, data set access, and feasibility. However, data sets are becoming more readily available to the public. Technological advancements have made the usage of these tools easier and more available. More people are cross-educated with GIS, meaning more and more people utilize GIS capabil-

ities in a wide variety of fields. These trends will lead to more rural decision makers having access to the knowledge obtained through the use of GIS and decision support tools.

Cross References

- ► Decision-Making Effectiveness with GIS
- ► Distributed Hydrologic Modeling
- ► Environmental Planning and Simulation Tools
- ► Hurricane Wind Fields, Multivariate Modeling
- ► Hydrologic Impacts, Spatial Simulation

Recommended Reading

1. Van Liew, M.W., Garbrecht, J.D., Arnold, J.G.: Simulation of the impacts of flood retarding structures on streamflow for a watershed in southwestern Oklahoma under dry, average, and wet climatic conditions. J. Soil Water Conserv. **58**(6), 340–348 (2003)
2. Schär, C., Vasilina, L., Pertziger, F., Dirren, S.: Seasonal runoff forecasting using precipitation from meteorological data assimilation systems. J. Hydrometeorol. **5**(5), 959–973 (2004)
3. Fohere, N., Haverkamp, S., Frede, H.G.: Assissment of the effects of land use patterns on hydrologic landscape functions: development of sustainable land use concepts for low mountain range areas. Hydrol. Process. **19**(3), 659–672 (2005)
4. Cau, P., Cadeddu, A., Gallo, C., Lecca, G., Marrocu, M.: Estimating Available Water Resources of the Sardinian Island Using the SWAT Model. SWAT2003 2nd International SWAT Conference Proceedings, Bari, Italy. Retrieved July 1, (2004) from SWAT Proceedings Web site: http://www.brc.tamus.edu/swat/2ndswatconf/2ndswatconfproceeding.pdf (64–70) (2003)
5. Melesse, Assefa M., Graham, Wendy D., Jordan, Jonathan D.: Spatially Distributed Watershed Mapping and Modeling: GIS-Based Storm Runoff Response and Hydrograph Analysis: Part 2. J. Spat. Hydro. **3** (2003)
6. Whittaker, G.: Application of SWAT in the evaluation of salmon habitat remediation policy. Hydrol. Process. **19**(3), 839–848 (2005)
7. Echardt, K., Ulbrich, U.: Potential impacts of climate change on groundwater recharge and streamflow in a central European low mountain range. J. Hydrol. **284**(1–4), 244–252 (2003)
8. Grabow, D.: Overland Flooding Risk Assessment in Rural Communities. Master's Thesis, University of North Dakota (2004)
9. Chaplot, V., Saleh, A., Jaynes, D.B., Arnold, J.: Predicting water, sediment, and NO_3-N Loads Under Scenarios of land-use and management practices in a flat watershed. Water, Air, Soil Pollut. **154**(1–4), 271–293 (2004)
10. Gutzler, D., Nims, J.: Interannual Variability of water demand and summer climate in Albuquerque, New Mexico. J. Appl. Meteorology. **44**, 1777–1787 (2005)
11. Sun, H., Cornish, P.S.: Estimating shallow groundwater recharge in the headwaters of the Liverpool Plains using SWAT. Hydrol. Process. **19**(3), 795–807 (2005)

Rural Hydrology

- ► Rural Hydrologic Decision Support