Count Queries in Probabilistic Spatio-Temporal Knowledge Bases with Capacity Constraints

John Grant[1] Cristian Molinaro[2] Francesco Parisi[2]

[1]Department of Computer Science and UMIACS,
University of Maryland, College Park, USA,
email: grant@cs.umd.edu
[2]Department of Informatics, Modeling, Electronics and System Engineering,
DIMES Department, University of Calabria, Italy,
email:{cmolinaro,fparisi}@dimes.unical.it

14th European Conference on Symbolic and
Quantitative Approaches to Reasoning with Uncertainty (ECSQARU 2017)
Lugano, Switzerland
July 10–14, 2017

Tracking moving objects (1/2)

- Tracking moving objects is fundamental in several application contexts (e.g. environment protection, product traceability, traffic monitoring, mobile tourist guides, analysis of animal behavior, etc.)

http://www.merl.com/publications/TR2008-010

http://www.edimax.com/au/

http://iris.usc.edu/people/medioni/curren t_research.html

http://www.i3b.org/content/wildlife-behavior

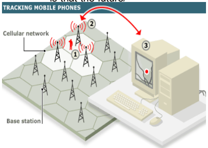
http://www.science20.com/news_articles/german_researc h_center_artificial_intelligence_smart_eye_tracking_glass

Tracking moving objects (2/2)

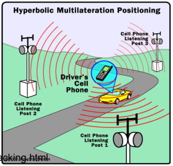
- Location estimation techniques have limited accuracy and precision
  - limitations of technologies used (e.g. GPS, Cellular networks, WiFi, Bluetooth, RFID, etc.)
  - limitations of the estimation methods (e.g., proximity to antennas, triangulation, signal strength sample map, dead reckoning, etc.)

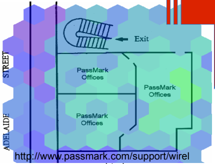
http://www.nitrobahn.com/conceptz/self-driving-cars-is-that-the-future/

http://www.ayantra.com/traffic-control-monitoring.html



http://www.gksoft.in/2014/07/mobile-phone-tracking.html





http://www.passmark.com/support/wireless_coverage_map.html

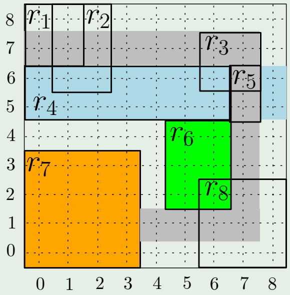
object inside a region at a time with uncertain probability

SPOT framework

- SPOT: a declarative framework for the representation and processing of probabilistic spatio-temporal data with uncertain probabilities [Parker, Subrahmanian, Grant. TKDE '07]
- A SPOT database is a set of atoms $loc(id, r, t)[\ell, u]$
- $loc(id, r, t)[\ell, u]$ means that "object id is/was/will be inside region r at time t with probability in the interval $[\ell, u]$".

Example

$loc(id_1, r_7, 0)[.9, 1]$
$loc(id_1, r_8, 1)[.6, .8]$
$loc(id_1, r_3, 2)[.4, .6]$
$loc(id_2, r_7, 0)[.9, 1]$
$loc(id_2, r_5, 1)[.4, .8]$
$loc(id_2, r_2, 2)[.4, .6]$
$loc(id_2, r_1, 2)[.3, .6]$
$loc(id_3, r_7, 0)[.9, 1]$
$loc(id_3, r_7, 1)[.9, 1]$



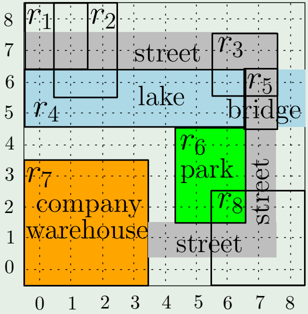
Atoms' region	bottom-left endpoint	top-right endpoint
r_1	(0, 7)	(1, 8)
r_2	(1, 6)	(2, 8)
r_3	(6, 6)	(7, 7)
r_4	(0, 5)	(6, 6)
r_5	(7, 5)	(7, 6)
r_6	(5, 2)	(6, 4)
r_7	(0, 0)	(3, 3)
r_8	(6, 0)	(8, 2)

Limits of SPOT DBs

- Although PST atoms express much useful information, they **cannot express** additional knowledge such as constraints on how many objects are allowed in a region, i.e., ***capacity constraints***

Example

1) There cannot be more than one truck on the bridge (region r_5) at any time
2) The number of trucks in the company warehouse is between 1 and 3 at any time between 0 and 1
3) No truck can be in the lake or the botanic park at any time point

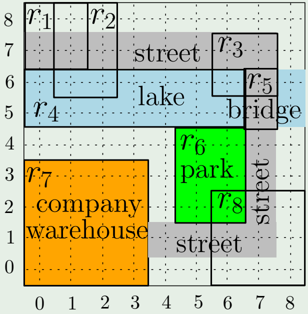


Limits of SPOT DBs

- Although PST atoms express much useful information, they **cannot express** additional knowledge such as constraints on how many objects are allowed in a region, i.e., ***capacity constraints***

Example

1) There cannot be more than one truck on the bridge (region r_5) at any time
2) The number of trucks in the company warehouse is between 1 and 3 at any time between 0 and 1
3) No truck can be in the lake or the botanic park at any time point

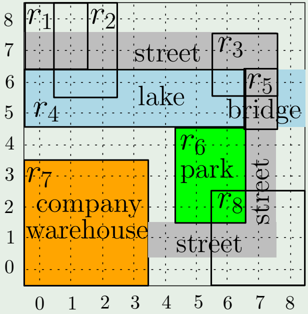


Limits of SPOT DBs

- Although PST atoms express much useful information, they **cannot express** additional knowledge such as constraints on how many objects are allowed in a region, i.e., ***capacity constraints***

Example

1) There cannot be more than one truck on the bridge (region r_5) at any time
2) The number of trucks in the company warehouse is between 1 and 3 at any time between 0 and 1
3) No truck can be in the lake or the botanic park at any time point

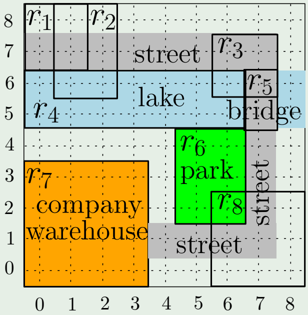


Limits of SPOT DBs

- Although PST atoms express much useful information, they **cannot express** additional knowledge such as constraints on how many objects are allowed in a region, i.e., ***capacity constraints***

Example

1) There cannot be more than one truck on the bridge (region r_5) at any time
2) The number of trucks in the company warehouse is between 1 and 3 at any time between 0 and 1
3) No truck can be in the lake or the botanic park at any time point



Probabilistic spatio-temporal KBs with capacity constraints

- We introduce probabilistic spatio-temporal (PST) knowledgebases (KB) consisting of

1) atomic statements, such as those representable in the SPOT framework
2) *capacity constraints*, each of them expressing lower- and/or upper-bounds on the number of objects that can be in a certain region.

- Formal semantics, in terms of worlds, interpretations, and models
- Complexity of checking consistency of PST KBs
  - NP-complete in general
  - Restricted classes of PST KBs for which the problem is in PTIME
- Count queries over (consistent) PST KBs:
  "How many objects are inside region q at time t?"
  - Formal semantics
  - Complexity
  - Show how checking consistency can be exploited for query answering

Probabilistic spatio-temporal KBs with capacity constraints

- We introduce probabilistic spatio-temporal (PST) knowledgebases (KB) consisting of

1) atomic statements, such as those representable in the SPOT framework
2) *capacity constraints*, each of them expressing lower- and/or upper-bounds on the number of objects that can be in a certain region.

- Formal semantics, in terms of worlds, interpretations, and models
- Complexity of checking consistency of PST KBs
  - NP-complete in general
  - Restricted classes of PST KBs for which the problem is in PTIME
- Count queries over (consistent) PST KBs:
  "How many objects are inside region q at time t?"
  - Formal semantics
  - Complexity
  - Show how checking consistency can be exploited for query answering

Probabilistic spatio-temporal KBs with capacity constraints

- We introduce probabilistic spatio-temporal (PST) knowledgebases (KB) consisting of

1) atomic statements, such as those representable in the SPOT framework
2) *capacity constraints*, each of them expressing lower- and/or upper-bounds on the number of objects that can be in a certain region.

- Formal semantics, in terms of worlds, interpretations, and models
- Complexity of checking consistency of PST KBs
  - NP-complete in general
  - Restricted classes of PST KBs for which the problem is in PTIME
- Count queries over (consistent) PST KBs:
  "How many objects are inside region q at time t?"
  - Formal semantics
  - Complexity
  - Show how checking consistency can be exploited for query answering

Outline

PST atoms

- We assume a finite set *ID* of *object ids*, a finite set *Space* of *spatial points*.
- A non-empty subset of *Space* is called a *region*.
- Arbitrarily large but fixed size window of time $T = [0, 1, \ldots, tmax]$.

A *spatio-temporal atom* (*st-atom*) is an expression of the form $loc(id, r, t)$, where $id \in ID$, $\emptyset \subsetneq r \subseteq Space$, and $t \in T$.

Definition (PST atom – SPOT atom in the previous framework)

A PST *atom* is an st-atom $loc(id, r, t)$ annotated with a probability interval $[\ell, u] \subseteq [0, 1]$ – denoted as $loc(id, r, t)[\ell, u]$.

- $loc(id, r, t)[\ell, u]$ says that object *id* is/was/will be inside region r at time t with probability in the interval $[\ell, u]$
- A SPOT database is a finite set of PST atoms. We extend the SPOT framework to consider capacity constraints.

PST atoms

- We assume a finite set *ID* of *object ids*, a finite set *Space* of *spatial points*.
- A non-empty subset of *Space* is called a *region*.
- Arbitrarily large but fixed size window of time $T = [0, 1, \ldots, tmax]$.

A *spatio-temporal atom* (*st-atom*) is an expression of the form $loc(id, r, t)$, where $id \in ID$, $\emptyset \subsetneq r \subseteq Space$, and $t \in T$.

Definition (PST atom – SPOT atom in the previous framework)

A PST *atom* is an st-atom $loc(id, r, t)$ annotated with a probability interval $[\ell, u] \subseteq [0, 1]$ – denoted as $loc(id, r, t)[\ell, u]$.

- $loc(id, r, t)[\ell, u]$ says that object *id* is/was/will be inside region r at time t with probability in the interval $[\ell, u]$
- A SPOT database is a finite set of PST atoms. We extend the SPOT framework to consider capacity constraints.

Capacity Constraints

Definition (Capacity constraint)

A *capacity constraint* is an expression of the form $capacity(r, k_1, k_2, t)$, where r is a region, k_1 and k_2 are two integers such that $0 \leq k_1 \leq k_2 \leq |ID|$, and t is a time point in T.

Example

1) $\kappa_{1,t} = capacity(r_5, 0, 1, t)$ with $t \in [0, 2]$,
there cannot be more than one truck on the bridge (region r_5) at any time between 0 and 2
2) $\kappa_{2,t} = capacity(r_7, 1, 3, t)$, with $t \in [0, 1]$,
the number of trucks in the company warehouse (region r_7) is between 1 and 3 at any time between 0 and 1
3) $\kappa_{3,t} = capacity(r_4, 0, 0, t)$ and
$\kappa_{4,t} = capacity(r_6, 0, 0, t)$, with $t \in [0, 2]$,
no truck can be in the lake (region r_4) or the botanic park (region r_6) at any time point (assuming $tmax = 2$)

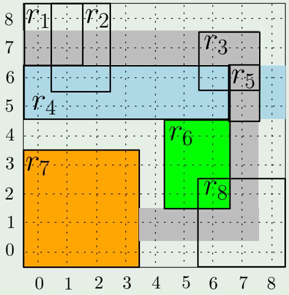
PST knowledge base

Definition (PST knowledge base)

A PST knowledge base is a pair $\langle \mathcal{A}, \mathcal{C} \rangle$, where $\mathcal{A}$ is a finite set of PST atoms and $\mathcal{C}$ is a finite set of capacity constraints.

Example

$loc(id_1, r_7, 0)[.9, 1]$
$loc(id_1, r_8, 1)[.6, .8]$
$loc(id_1, r_3, 2)[.4, .6]$
$loc(id_2, r_7, 0)[.9, 1]$
$loc(id_2, r_5, 1)[.4, .8]$
$loc(id_2, r_2, 2)[.4, .6]$
$loc(id_2, r_1, 2)[.3, .6]$
$loc(id_3, r_7, 0)[.9, 1]$
$loc(id_3, r_7, 1)[.9, 1]$



$\kappa_{1,t} = capacity(r_5, 0, 1, t)$
$t \in [0, 2]$
$\kappa_{2,t} = capacity(r_7, 1, 3, t)$,
$t \in [0, 1]$,
$\kappa_{3,t} = capacity(r_4, 0, 0, t)$,
$\kappa_{4,t} = capacity(r_6, 0, 0, t)$,
$t \in [0, 2]$,

World

- A world specifies a possible trajectory for each object $id \in ID$ (i.e., says where in *Space* object *id* was/is/will be at each time $t \in T$)

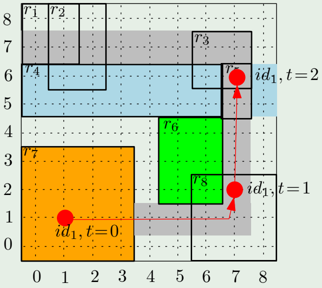
Definition (World)

A world w is a function, $w : ID \times T \to Space$

Example

World w_1 describing the positions of id_1, id_2 and id_3 for time points in $[0, 2]$:

$$w_1(id_1, 0) = (1, 1)$$
$$w_1(id_1, 1) = (7, 2)$$
$$w_1(id_1, 2) = (7, 6)$$



World

- A world specifies a possible trajectory for each object $id \in ID$ (i.e., says where in *Space* object *id* was/is/will be at each time $t \in T$)

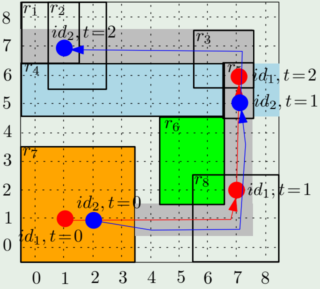
Definition (World)

A world w is a function, $w : ID \times T \to Space$

Example

World w_1 describing the positions of id_1, id_2 and id_3 for time points in $[0, 2]$:

$w_1(id_1, 0) = (1, 1)$
$w_1(id_1, 1) = (7, 2)$
$w_1(id_1, 2) = (7, 6)$
$w_1(id_2, 0) = (2, 1)$
$w_1(id_2, 1) = (7, 5)$
$w_1(id_2, 2) = (1, 7)$



World

- A world specifies a possible trajectory for each object $id \in ID$ (i.e., says where in *Space* object id was/is/will be at each time $t \in T$)

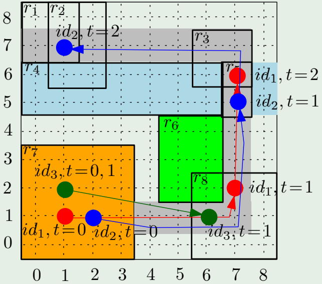
Definition (World)

A world w is a function, $w : ID \times T \rightarrow Space$

Example

World w_1 describing the positions of id_1, id_2 and id_3 for time points in $[0, 2]$:

$w_1(id_1, 0) = (1, 1)$
$w_1(id_1, 1) = (7, 2)$
$w_1(id_1, 2) = (7, 6)$
$w_1(id_2, 0) = (2, 1)$
$w_1(id_2, 1) = (7, 5)$
$w_1(id_2, 2) = (1, 7)$
$w_1(id_3, 0) = (1, 2)$
$w_1(id_3, 1) = (1, 2)$
$w_1(id_3, 2) = (6, 1)$



Satisfaction

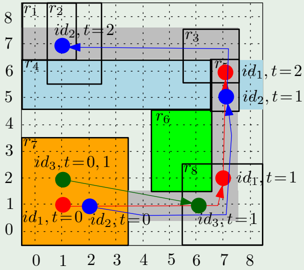
Definition (Satisfaction)

A world w *satisfies* an st-atom $a = loc(id, r, t)$, denoted $w \models a$, iff $w(id, t) \in r$. Moreover, w *satisfies* a capacity constraint $\kappa = capacity(r, k_1, k_2, t)$, denoted $w \models \kappa$, iff $k_1 \leq |\{id \in ID(\mathcal{K}) \mid w(id, t) \in r\}| \leq k_2$.

Example

World w_1 describing the positions of id_1, id_2 and id_3 for time points in $[0, 2]$:

$w_1(id_1, 0) = (1, 1)$
$w_1(id_1, 1) = (7, 2)$
$w_1(id_1, 2) = (7, 6)$
$w_1(id_2, 0) = (2, 1)$
$w_1(id_2, 1) = (7, 5)$
$w_1(id_2, 2) = (1, 7)$
$w_1(id_3, 0) = (1, 2)$
$w_1(id_3, 1) = (1, 2)$
$w_1(id_3, 2) = (6, 1)$



$w_1 \models loc(id_1, r_7, 0)$,
as $w_1(id_1, 0) = (1, 1) \in r_7$

$\forall t \in [0, 2],\ w_1 \models capacity(r_5, 0, 1, t)$
as $\{id \in ID(\mathcal{K}) \mid w_1(id, 0) \in r_5\} = \emptyset$
$\{id \in ID(\mathcal{K}) \mid w_1(id, 1) \in r_5\} = \{id_2\}$
$\{id \in ID(\mathcal{K}) \mid w_1(id, 2) \in r_5\} = \{id_1\}$

Interpretations

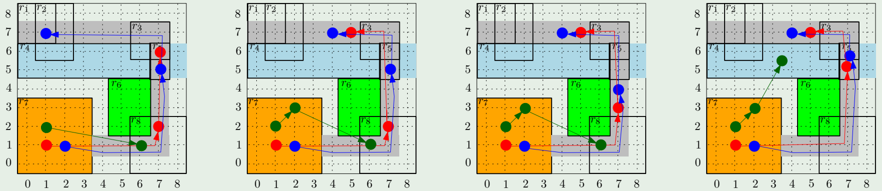
Definition (Interpretation)

An *interpretation* I for $\mathcal{K}$ is a PDF over the set $\mathcal{W}(\mathcal{K})$ of all worlds of $\mathcal{K}$.

- $I(w)$ is the probability that w describes the actual trajectories of all objects

Example (Interpretation I)

w_1 w_2 w_3 w_4



$I(w_1) = 0.6$ $I(w_2) = 0.2$ $I(w_3) = 0.2$ $I(w_4) = 0$

and all other words are assigned probability equal to zero by interpretation I

- Only the interpretations that are compatible with the information in $\mathcal{K}$ (PST atoms + Capacity constraints) are models

Models

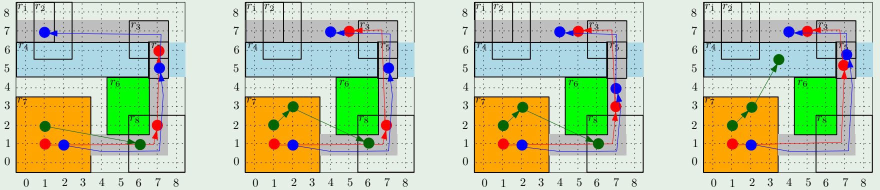
Definition (Model)

A model M for $\mathcal{K} = \langle \mathcal{A}, \mathcal{C} \rangle$ is an interpretation for $\mathcal{K}$ such that:

(i) $\forall\ loc(id, r, t)[\ell, u] \in \mathcal{A}, \left(\sum_{w \mid w \models loc(id,r,t)} M(w) \right) \in [\ell, u];$

(ii) $\forall\ \kappa \in \mathcal{C}, \sum_{w \mid w \not\models \kappa} M(w) = 0.$

Example (Model M)



$I(w_1) = 0.6$ $I(w_2) = 0.2$ $I(w_3) = 0.2$ $I(w_4) = 0$

- For atom $loc(id_1, r_7, 0)[.9, 1]$,
  $\sum_{w \mid w \models loc(id_1, r_7, 0)} M(w) = M(w_1) + M(w_2) + M(w_3) = 1 \in [.9, .1]$

Models

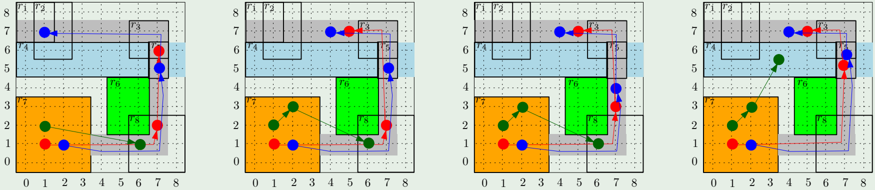
Definition (Model)

A model M for $\mathcal{K} = \langle \mathcal{A}, \mathcal{C} \rangle$ is an interpretation for $\mathcal{K}$ such that:

(i) $\forall\ loc(id, r, t)[\ell, u] \in \mathcal{A}, \left(\sum_{w \mid w \models loc(id,r,t)} M(w) \right) \in [\ell, u]$;

(ii) $\forall\ \kappa \in \mathcal{C}, \sum_{w \mid w \not\models \kappa} M(w) = 0.$

Example (Model M)



$I(w_1) = 0.6$ $I(w_2) = 0.2$ $I(w_3) = 0.2$ $I(w_4) = 0$

- $M(w_4) = 0$ since w_4 violates the constraint $\kappa_{1,1} = capacity(r_5, 0, 1, t)$, as there are 2 trucks on the bridge at time 1 according w_4

Consistency

- The set of models for $\mathcal{K}$ will be denoted as $\mathbf{M}(\mathcal{K})$.
- $\mathcal{K}$ is *consistent* iff there exists a model for it (i.e., $\mathbf{M}(\mathcal{K}) \neq \emptyset$)
- PST KB of our running example is consistent

Outline

Complexity

Theorem

Deciding whether a PST *KB* $\mathcal{K}$ *is consistent is NP-complete.*

- Membership: deciding whether $\mathcal{K}$ is consistent corresponds to checking the feasibility of

$$LP(\mathcal{K}) := \begin{cases} (1) & \forall\ loc(id, r, t)[\ell, u] \in \mathcal{A}, \\ & \quad (a)\ \ \ell \leq \sum_{w_i | w_i \models loc(id,r,t)} v_i \\ & \quad (b)\ \ \sum_{w_i | w_i \models loc(id,r,t)} v_i \leq u \\ (2) & \forall \kappa \in \mathcal{C},\ \sum_{w_i \mid w_i \not\models \kappa} v_i = 0 \\ (3) & \sum_{w_i \mid w_i \in \mathcal{W}(\mathcal{K})} v_i = 1 \\ (4) & \forall w_i \in \mathcal{W}(\mathcal{K}),\ \ v_i \geq 0 \end{cases}$$

- v_i represents probability $M(w_i)$ assigned to $w_i \in \mathcal{W}(\mathcal{K})$ by $M \in \mathbf{M}(\mathcal{K})$
- Exponential number of variables v_i (i.e., $|\mathcal{W}(\mathcal{K})| = |Space|^{|ID| \cdot |T|}$)

Complexity

Theorem

Deciding whether a PST *KB* $\mathcal{K}$ *is consistent is NP-complete.*

- Membership: deciding whether $\mathcal{K}$ is consistent corresponds to checking the feasibility of

$$LP(\mathcal{K}) := \begin{cases} (1) & \forall\ loc(id,r,t)[\ell,u] \in \mathcal{A}, \\ & \quad (a)\ \ \ell \leq \sum\limits_{w_i | w_i \models loc(id,r,t)} v_i \\ & \quad (b)\ \ \sum\limits_{w_i | w_i \models loc(id,r,t)} v_i \leq u \\ (2) & \forall \kappa \in \mathcal{C}, \sum\limits_{w_i \,|\, w_i \not\models \kappa} v_i = 0 \\ (3) & \sum\limits_{w_i \,|\, w_i \in \mathcal{W}(\mathcal{K})} v_i = 1 \\ (4) & \forall w_i \in \mathcal{W}(\mathcal{K}),\ v_i \geq 0 \end{cases}$$

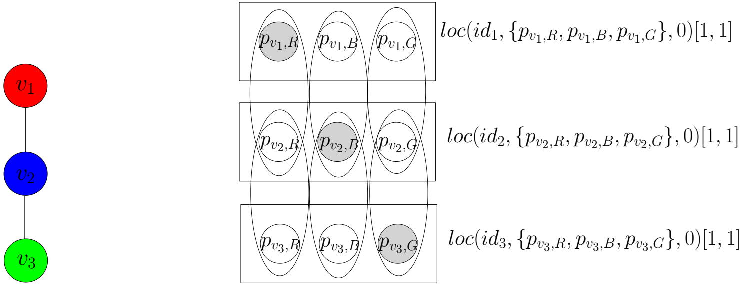
- v_i represents probability $M(w_i)$ assigned to $w_i \in \mathcal{W}(\mathcal{K})$ by $M \in \mathbf{M}(\mathcal{K})$
- Exponential number of variables v_i (i.e., $|\mathcal{W}(\mathcal{K})| = |Space|^{|ID|\cdot|T|}$)

Membership in NP

- It can be shown that $LP(\mathcal{K})$ is feasible iff there is a solution for $LP(\mathcal{K})$ consisting of at most $2 \cdot |\mathcal{A}| + |\mathcal{C}| + 1$ non-zero variables (it follows from a well-known result on the size of solutions of linear programming problems [Papadimitriou, Steiglitz '82])
- Guess an assignment s' consisting of $2 \cdot |\mathcal{A}| + |\mathcal{C}| + 1$ pairs $\langle v_i, \text{value of } v_i \rangle$,
- Check in polynomial time whether s' is a solution of $LP^*(\mathcal{K})$, obtained from $LP(\mathcal{K})$ by keeping in it only the variables in s'
- If s' is a solution of $LP^*(\mathcal{K})$, then $LP(\mathcal{K})$ is feasible

NP-hardness

- Reduction from 3-COLORING problem
- Given $G = \langle V, E \rangle$, use 3 points $p_{v,R}$, $p_{v,G}$,$p_{v,B}$ in *Space* for each $v \in V$
- PST atom $loc(id_v, \{p_{v,R}, p_{v,G}, p_{v,B}\}, 0)[1, 1]$ for each vertex $v \in V$
- $capacity(\{p_{i,col}, p_{j,col}\}, 0, 1, 0)$ for each edge $(i, j) \in E$ and color $col \in \{R, G, B\}$



- G is 3-colorable iff $\mathcal{K}$ is consistent

Tractable cases

- Capacity constraints allowing no objects in some regions (e.g., there cannot be trucks in the lake)

Theorem

Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{C} \rangle$ *be a* PST *KB. If* $\mathcal{C}$ *consists of capacity constraints of the form* $capacity(r, 0, 0, t)$*, then checking whether* $\mathcal{K}$ *is consistent is in PTIME.*

- Proof hint: it can be reduced to checking consistency of a KB having no capacity constraints, which is in PTIME [Parker, Subrahmanian, Grant. TKDE '07]
- $capacity(r, 0, 0, t)$ can be translated into the set of additional atoms $\forall id \in ID, loc(id, Space \setminus r, t)[1, 1]$

Sufficient conditions for checking consistency (1/2)

- Upper bounds of all PST atoms is 1 and
- regions in different capacity constraints are disjoint

Theorem

Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{C} \rangle$ be a PST *KB that satisfies the following conditions:*

- *$\mathcal{A}$ consists of* PST *atoms of the form $loc(id, r, t)[\ell, 1]$ and there are no two distinct* PST *atoms in $\mathcal{A}$ for the same object id and time point t, and*
- *for every time point t, every pair of distinct capacity constraints $capacity(r, k_1, k_2, t)$ and $capacity(r', k'_1, k'_2, t)$ in $\mathcal{C}$ is such that $r \cap r' = \emptyset$.*

Deciding if there exists a world $w \in \mathcal{W}(\mathcal{K})$ s.t. (i) $w \models \mathcal{C}$ and (ii) $w(id, t) \in r$ for every $loc(id, r, t)[\ell, 1]$ in $\mathcal{A}$ with $\ell > 0$, is in PTIME. If such a world exists, then $\mathcal{K}$ is consistent.

- reduction to the problem of deciding if a flow network admits a feasible circulation

Sufficient conditions for checking consistency (2/2)

- A PST KB $\langle \mathcal{A}, \mathcal{C} \rangle$ is called *simple* iff for every time point $t \in T$, there is at most one capacity constraint of the form $capacity(r, k_1, k_2, t)$ in $\mathcal{C}$

Theorem

Let $\mathcal{K} = \langle \mathcal{A}, \mathcal{C} \rangle$ be a simple PST *KB. If $\langle \mathcal{A}, \emptyset \rangle$ is consistent and, for every $capacity(r, k_1, k_2, t) \in \mathcal{C}$, $[z, Z] \subseteq [k_1, k_2]$, where*

$$z = \min_{M \in \mathbf{M}(\langle \mathcal{A}, \emptyset \rangle)} |\{id \mid id \in ID \wedge \left(\sum_{w \mid w(id,t) \in r} M(w) \right) = 1\}|,$$

$$Z = \max_{M \in \mathbf{M}(\langle \mathcal{A}, \emptyset \rangle)} |\{id \mid id \in ID \wedge \left(\sum_{w \mid w(id,t) \in r} M(w) \right) \neq 0\}|,$$

then $\mathcal{K}$ is consistent. Checking consistency under such conditions is in PTIME.

- Computing $[z, Z]$ is in PTIME [Grant, Molinaro, Parisi. SUM 2013]

Outline

Syntax and semantics

- $Count(q, t)$ asks "How many objects are inside region q at time t?"
- Ranking answer: set of pairs $\langle i, [\ell_i, u_i]\rangle$ where
  - i is the number of objects that may be in q at time t
  - ℓ_i and u_i are the minimum and maximum probabilities of having exactly i objects in q at a time t over all models
- For a given model M, the probability of having exactly i objects in a region q at a time point t w.r.t. M is $Prob_M(q, i, t) = \sum_{w \mid w \models capacity(q,i,t)} M(w)$

Definition (Ranking Answer)

The ranking answer to a count query $Q = Count(q, t)$ w.r.t. $\mathcal{K}$ is:

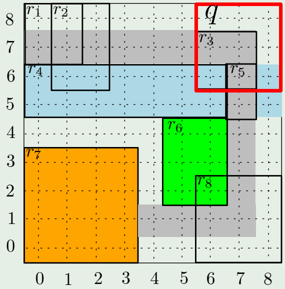
$$Q(\mathcal{K}) = \{\langle i, [\ell_i, u_i]\rangle \mid \quad 0 \le i \le |ID| \wedge \quad \ell_i = \min_{M \in \mathbf{M}(\mathcal{K})} Prob_M(q, i, t) \wedge$$
$$u_i = \max_{M \in \mathbf{M}(\mathcal{K})} Prob_M(q, i, t)\}.$$

Example

Example

- How many trucks are in q (the red square) at time 2?

$loc(id_1, r_7, 0)[.9, 1]$
$loc(id_1, r_8, 1)[.6, .8]$
$loc(id_1, r_3, 2)[.4, .6]$
$loc(id_2, r_7, 0)[.9, 1]$
$loc(id_2, r_5, 1)[.4, .8]$
$loc(id_2, r_2, 2)[.4, .6]$
$loc(id_2, r_1, 2)[.3, .6]$
$loc(id_3, r_7, 0)[.9, 1]$
$loc(id_3, r_7, 1)[.9, 1]$



$\kappa_{1,t} = capacity(r_5, 0, 1, t)$
$t \in [0, 2]$
$\kappa_{2,t} = capacity(r_7, 1, 3, t)$,
$t \in [0, 1]$,
$\kappa_{3,t} = capacity(r_4, 0, 0, t)$,
$\kappa_{4,t} = capacity(r_6, 0, 0, t)$,
$t \in [0, 2]$,

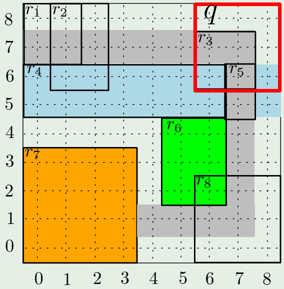
- Ranking answer $Q(\mathcal{K}) = \{\langle 0, [.4, .6]\rangle, \langle 1, [.4, 1]\rangle, \langle 2, [0, .3]\rangle, \langle 3, [0, .1]\rangle\}$
- For instance, $\langle 1, [.4, 1]\rangle$ says that the probability of having exactly one object in q at time 2 is between .4 and 1.

Example

Example

- How many trucks are in q (the red square) at time 2?

$loc(id_1, r_7, 0)[.9, 1]$
$loc(id_1, r_8, 1)[.6, .8]$
$loc(id_1, r_3, 2)[.4, .6]$
$loc(id_2, r_7, 0)[.9, 1]$
$loc(id_2, r_5, 1)[.4, .8]$
$loc(id_2, r_2, 2)[.4, .6]$
$loc(id_2, r_1, 2)[.3, .6]$
$loc(id_3, r_7, 0)[.9, 1]$
$loc(id_3, r_7, 1)[.9, 1]$



$\kappa_{1,t} = capacity(r_5, 0, 1, t)$
$t \in [0, 2]$
$\kappa_{2,t} = capacity(r_7, 1, 3, t)$,
$t \in [0, 1]$,
$\kappa_{3,t} = capacity(r_4, 0, 0, t)$,
$\kappa_{4,t} = capacity(r_6, 0, 0, t)$,
$t \in [0, 2]$,

- Ranking answer $Q(\mathcal{K}) = \{\langle 0, [.4, .6]\rangle, \langle 1, [.4, 1]\rangle, \langle 2, [0, .3]\rangle, \langle 3, [0, .1]\rangle\}$
- For instance, $\langle 1, [.4, 1]\rangle$ says that the probability of having exactly one object in q at time 2 is between .4 and 1.

Complexity

Theorem

Computing $Q(\mathcal{K})$ is $FP^{NP[\log n]}$-hard.

- Reduction to our problem from the $FP^{NP[\log n]}$-hard problem CLIQUE SIZE: determine the size σ of the largest clique of a graph $G = \langle V, E\rangle$
- Proof hint: An id id_v and two spatial points $p_{v,in}, p_{v,out}$ for each $v \in V$
- PST atom saying that id_v must be at one of the two points $p_{v,in}, p_{v,out}$
- $capacity(\{p_{i,in}, p_{j,in}\}, 0, 1, 0)$ for each $(i, j) \in (V \times V) \setminus E$ saying that no more than one object can be in the region consisting of two *in* points associated with a pair of vertices *not* connected by an edge
- $Q = Count(\{p_{1,in}, \ldots, p_{n,in}\}, 0)$.
- The size of the largest clique of G is σ iff $Q(\mathcal{K}) = \{\langle i, [0, 1]\rangle \mid 0 \leq i \leq \sigma\} \cup \{\langle i, [0, 0]\rangle \mid \sigma < i \leq |ID|\}$.

Using consistency checking to answering queries

- Solving some instances of the consistency check problem allows us to answer some count queries
- Given $\mathcal{K} = \langle \mathcal{A}, \mathcal{C} \rangle$, we check consistency of $\mathcal{K}' = \langle \mathcal{A}, \mathcal{C}' \rangle$ to get the answers

Proposition

Let $Q = Count(q, t)$ and $\mathcal{K} = \langle \mathcal{A}, \mathcal{C} \rangle$.

- *If $\mathcal{K}' = \langle \mathcal{A}, \mathcal{C} \cup \{capacity(q, k_1, k_2, t)\} \rangle$ is consistent, then $\ell_i = 0$ in $Q(\mathcal{K})$ for all i such that $i < k_1$ or $i > k_2$.*
- *If $\mathcal{K}' = \langle \mathcal{A}, \mathcal{C} \cup \{capacity(Space \setminus q, k_1, k_2, t)\} \rangle$ is consistent, then $u_i = 1$ in $Q(\mathcal{K})$ for all $i \in [|ID| - k_2, |ID| - k_1]$.*

Outline

Conclusions and future work

- A declarative language suitable in many applications dealing with uncertain spatio-temporal data
- Capacity constraints allow us to model semantic information commonly arising in practice
- We have investigated the complexity of checking consistency and answering count queries
- Intractable in general, but tractable approaches for restricted cases
- Further issues that we plan to investigate:
  - other tractable cases
  - the interaction between capacity constraints and the universal denial constraints proposed in [Parisi, Grant JAIR 2016] to get a unified approach that allows for a wide range of constraints to be expressed
  - the problems of repairing and querying inconsistent PST KBs with capacity constraints (following [Parisi, Grant IJAR 2017] where the problem of restoring consistency of PST KBs without integrity constraints has been explored)

Conclusions and future work

- A declarative language suitable in many applications dealing with uncertain spatio-temporal data
- Capacity constraints allow us to model semantic information commonly arising in practice
- We have investigated the complexity of checking consistency and answering count queries
- Intractable in general, but tractable approaches for restricted cases
- Further issues that we plan to investigate:
    - other tractable cases
    - the interaction between capacity constraints and the universal denial constraints proposed in [Parisi, Grant JAIR 2016] to get a unified approach that allows for a wide range of constraints to be expressed
    - the problems of repairing and querying inconsistent PST KBs with capacity constraints (following [Parisi, Grant IJAR 2017] where the problem of restoring consistency of PST KBs without integrity constraints has been explored)

Thank you!

... any question?

Location estimation techniques

- Location estimation techniques build on different technologies (e.g. GPS, Cellular networks, WLAN, Bluetooth, RFID, etc.)
  - proximity techniques derive the location of an object w.r.t. its vicinity to antennas
  - triangulation uses the triangle geometry to compute locations of an object.
  - scene analysis techniques (e.g. fingerprinting technique) involve examination and matching a video/image or electromagnetic characteristics viewed/sensed from an object
  - Dead reckoning techniques provide estimation of the location of an object based on the last known position, assuming that the direction of motion and either the velocity of the target object or the travelled distance are known
  - hybrid techniques
- Several sources of spatial temporal information (e.g. GPS, Cellular networks, WLAN, Wi-Fi), Bluetooth, Zigbee, Ultra-wideband (UWB), and Radio-frequency identification (RFID), or infrared (IR)

Selected References

Parker, A., Subrahmanian, V.S., Grant, J.
A logical formulation of probabilistic spatial databases.
IEEE TKDE, pp. 1541–1556, 2007.

John Grant, Cristian Molinaro, Francesco Parisi
Aggregate Count Queries in Probabilistic Spatio-temporal Databases.
Int. Conf. on Scalable Uncertainty Management (SUM), pp. 255-268, 2013.

Francesco Parisi, John Grant,
Knowledge Representation in Probabilistic Spatio-Temporal Knowledge Bases
J. Artif. Intell. Res., pp. 743-798, 2016

Francesco Parisi, John Grant,
On repairing and querying inconsistent probabilistic spatio-temporal databases
Int. J. Approx. Reasoning, pp. 41-74, 2017

Papadimitriou, C.H., Steiglitz, K.,
Combinatorial optimization: algorithms and complexity.
Prentice-Hall, Inc., 1982.