Inconsistent Databases

Querying Relational Databases

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	EDI	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

Querying Relational Databases

List the airlines that fly directly from London to Glasgow

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	EDI	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

⇓

{U2}

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)

Semantic Information About the Data

inconsistency!!!

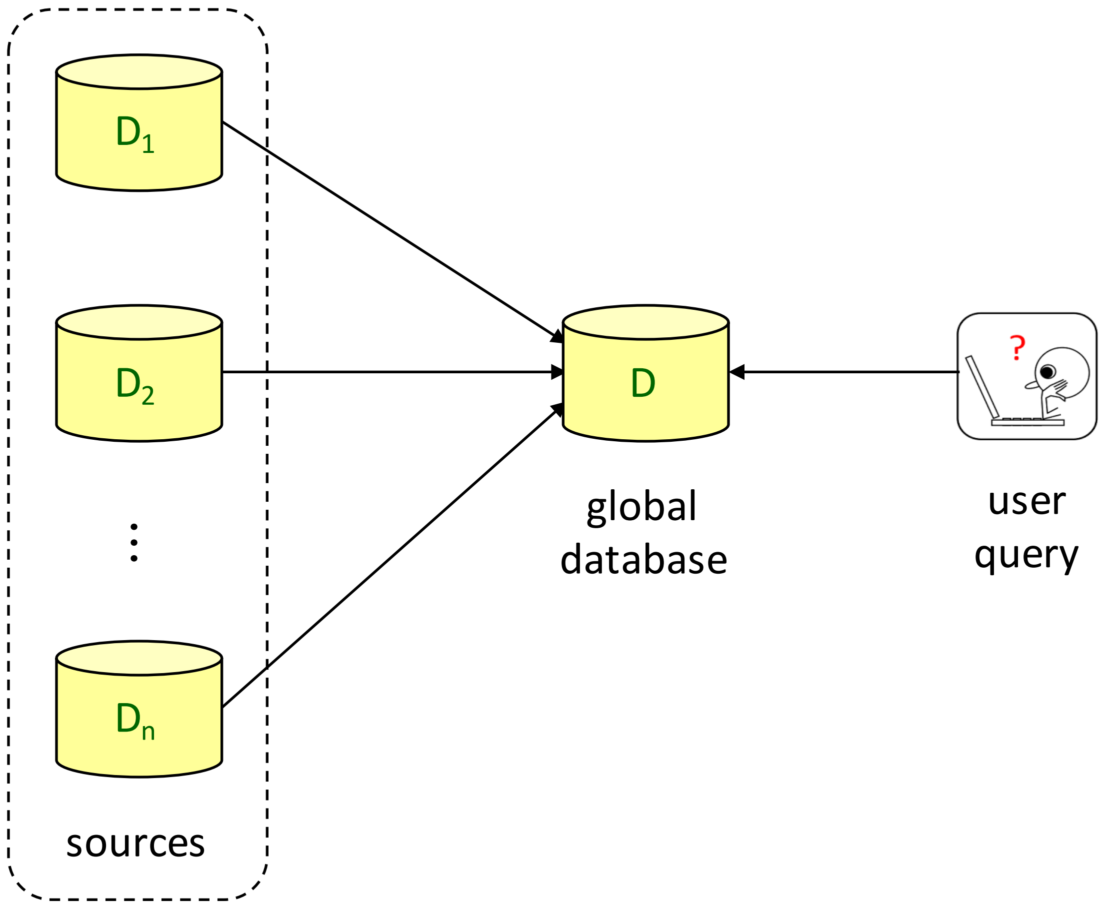
Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	EDI	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

The code uniquely determines the airport

Main Source of Inconsistency

data is coming from several conflicting sources



Querying Relational Databases

List the airlines that fly directly from London to Glasgow

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	EDI	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

{U2} ?

Q(z) :- Airport(x,London), Airport(y,Glasgow), Flight(x,y,z)

Integrity Constraints

inconsistency!!!

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	EDI	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

$$\text{Key(Airport)} = \{1\} \quad \equiv \quad \forall x \forall y \, (\text{Airport}(x,y) \land \text{Airport}(x,z) \rightarrow y = z)$$

Primary Keys

at most one key per relation

inconsistency!!!

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	EDI	VIE	OS

Airport	<u>code</u>	city
	VIE	Vienna
	LHR	London
	LGW	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

$$\text{Key(Airport)} = \{1\} \quad \equiv \quad \forall x \forall y \, (\text{Airport}(x,y) \land \text{Airport}(x,z) \rightarrow y = z)$$

Primary Keys

at most one key per relation

- Consider a database D, and a primary key σ : $Key(R) = \{i_1,\ldots,i_n\}$

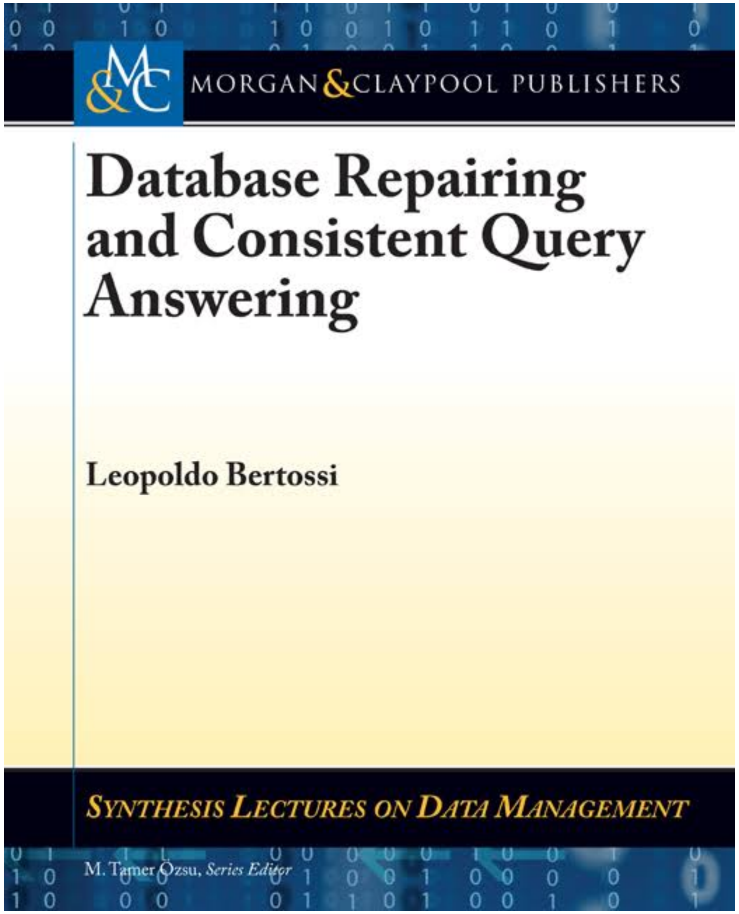
  We say that D satisfies σ if, for every two atoms $R(a_1,\ldots, a_m)$ and $R(b_1,\ldots,b_m)$ in D

  such that $a_{i_1},\ldots,a_{i_n} = b_{i_1},\ldots,b_{i_n}$, it holds that $a_1,\ldots, a_m = b_1,\ldots,b_m$

- D satisfies a set of primary keys Σ, denoted $D \models \Sigma$, if D satisfies every key in Σ

  In this case we say that D is **consistent** w.r.t. Σ; otherwise, D is **inconsistent** w.r.t. Σ

Consistent Query Answering (CQA)

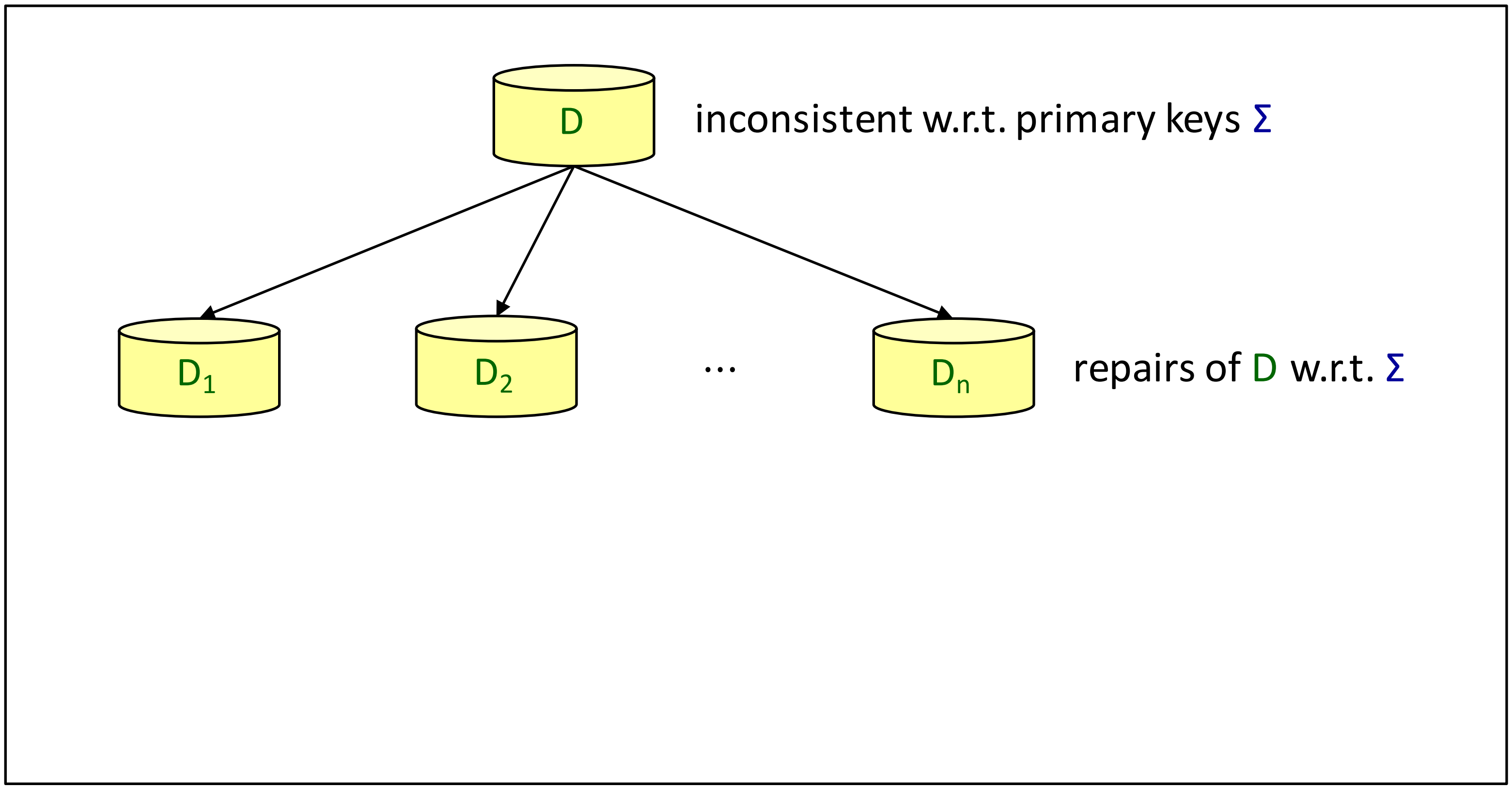


find meaningful answers to queries

when databases are inconsistent

Key Elements of CQA

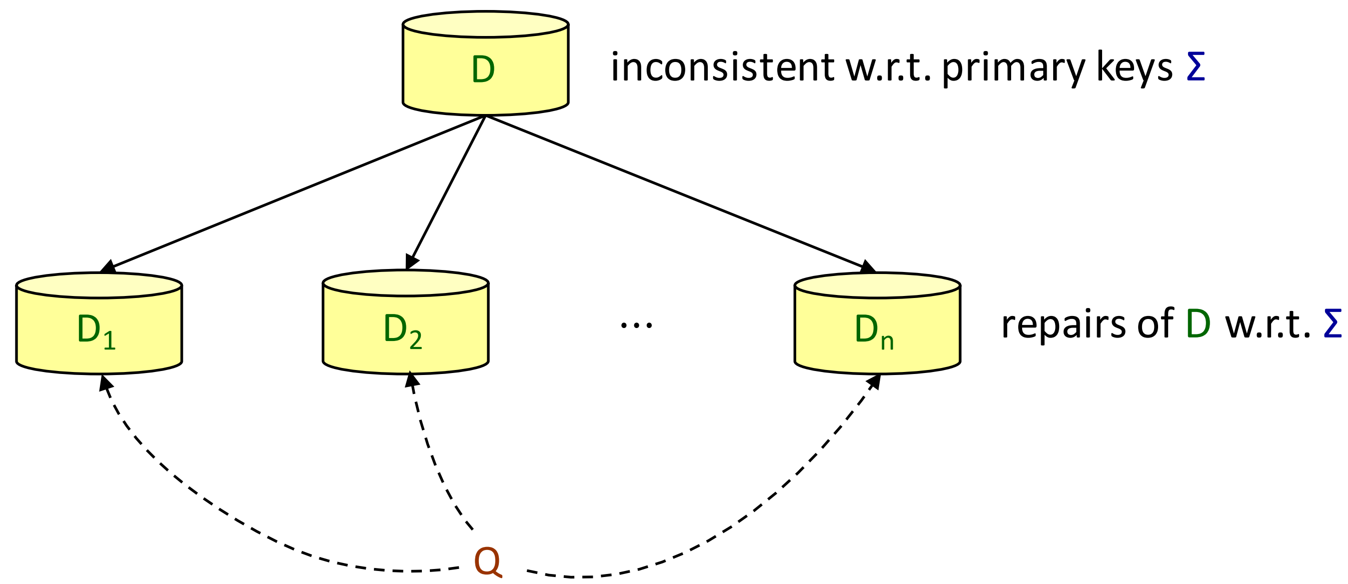
Repairs - consistent databases whose difference with D is “minimal”



Key Elements of CQA

Repairs - consistent databases whose difference with D is "minimal"

Consistent answers - answers that are true in all repairs



$$\text{Answer}(Q,D,\Sigma) = Q(D_1) \cap \cdots \cap Q(D_n)$$

Key Elements of CQA

Consider a database D, and a set Σ of primary keys

A **repair** of D w.r.t. Σ is a database $D' \subseteq D$ such that the following conditions hold:

1. $D' \models \Sigma$
2. There is no $D'' \subseteq D$ such that $D'' \models \Sigma$ and $D' \subset D''$

$$\boxed{\text{Answer}(Q,D,\Sigma) \quad = \quad \bigcap_{R \,\in\, \text{repairs}(D,\Sigma)} Q(R)}$$

the set of repairs of D w.r.t. Σ

Repairs

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

Repair 1

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	GLA	Glasgow
	EDI	Edinburgh

Repair 2

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

Consistent Answers

List the airlines that fly directly from London to Glasgow

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	EDI	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

Consistent Answers

List the airlines that fly directly from London to Glasgow

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	$\boxed{\text{U2}}$
	EDI	VIE	OS

Airport	<u>code</u>	city
	VIE	Vienna
	LHR	London
	LGW	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

Repair 1: {U2}

Consistent Answers

List the airlines that fly directly from London to Glasgow

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	EDI	VIE	OS

Airport	code	city
	VIE	Vienna
	LHR	London
	LGW	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

Repair 1: {U2}

Repair 2: { }

Answer is empty

Consistent Query Answering

CQA(**L**)

Input: a database D, a set of primary keys Σ, a query Q/k ∈ **L**, a tuple of constants **t** ∈ **adom**(D)^k

Question: **t** ∈ Answer(Q,D,Σ)?

BCQA(**L**)

Input: a database D, a set of primary keys Σ, a Boolean query Q ∈ **L**

Question: is Answer(Q,D,Σ) non-empty?

Theorem: CQA(**L**) $\equiv_L$ BCQA(**L**), where **L** ∈ {**RA, DRC**, **TRC, CQ**}

($\equiv_L$ means logspace-equivalent)

Data Complexity of BCQA

input D, fixed Σ and Q

BCQA[Σ,Q](**L**)

Input: a database D

Question: is Answer(Q,D,Σ) non-empty?

Data Complexity of BCQA

Theorem: For **L** ∈ {**RA**, **DRC**, **TRC, CQ**}, BCQA[Σ,Q](**L**) is coNP-complete for a fixed set of primary keys Σ, and a query Q ∈ **L**.

Proof:

- Guess a repair R ∈ repairs(D,Σ), and check whether Q(R) is empty
- Reduction from 3-Colorability to the complement of BCQA

3-Colorability

3COL

Input: an undirected graph **G** = (V,E)

Question: is there a function $c : V \rightarrow \{R,G,B\}$ such that $(v,u) \in E \Rightarrow c(v) \neq c(u)$?

coNP-hardness

Given an undirected graph **G** = (V,E)

construct a database D such that, for some fixed Σ and Q, it holds that

G is 3-colorable iff Answer(Q,D,Σ) is empty

D = {Edge(u,v) : (u,v) ∈ E} ∪

{Color(v,r), Color(v,g), Color(v,b) : v ∈ V} ∪

{Bad(r,r), Bad(g,g), Bad(b,b)}

Σ = {Key(Color) = {1}}

Q :- Edge(x,y), Color(x,z), Color(y,w), Bad(z,w)

Lemma: G is 3-colorable iff there is R ∈ repairs(D,Σ) such that Q(R) is empty

Data Complexity of BCQA

Theorem: For **L** ∈ {**RA**, **DRC**, **TRC, CQ**}, BCQA[Σ,Q](**L**) is coNP-complete for a fixed set of primary keys Σ, and a query Q ∈ **L**.

Proof:

- Guess a repair R ∈ repairs(D,Σ), and check whether Q(R) is empty
- Reduction from 3-Colorability to the complement of BCQA

Tackle High Data Complexity

Two main research directions:

1. Isolate classes of queries (in fact, classes of CQs) for which the problem can be solved efficiently in data complexity

2. Provide data-efficient approximations

Consistent Answers

List the airlines that fly directly from London to Glasgow

Flight	origin	destination	airline
	VIE	LHR	BA
	LHR	EDI	BA
	LGW	GLA	U2
	EDI	VIE	OS

Airport	<u>code</u>	city
	VIE	Vienna
	LHR	London
	LGW	London
	LGW	Lilongwe
	GLA	Glasgow
	EDI	Edinburgh

Repair 1: {U2}

Repair 2: { }

Answer = { (U2,50%) }

Relative Frequency of a Boolean Query

$$RF(Q,D,\Sigma) = \frac{|\{R : R \in repairs(D,\Sigma) \text{ such that } Q(R) \text{ is non-empty}\}|}{|repairs(D,\Sigma)|}$$

Consistent Query Answering Revisited

RF-BCQA(**L**)

Input: a database D, a set of primary keys Σ, a Boolean query Q ∈ **L**

Output: RF(Q,D,Σ)

we can naturally talk about the data complexity the problem

RF-BCQA[Σ,Q](**L**) - input D, fixed Σ and Q

Data Complexity of BCQA

Theorem: For **L** ∈ {**RA**, **DRC**, **TRC, CQ**}, BCQA[Σ,Q](**L**) is $FP^{\#P}$-complete for a fixed set of primary keys Σ, and a query Q ∈ **L**.

- This essentially means that computing the relative frequency of a Boolean query is a hard problem, even for CQs
- The goal is to *efficiently approximate* the relative frequency

Efficient Approximations

fix a set of primary keys Σ, and a Boolean CQ Q

A **fully polynomial-time randomized approximation scheme** (FPRAS) for RF-BCQA[Σ,Q](**CQ**)

is a randomized algorithm **Approximation** that accepts as input

a database D, and numbers $\varepsilon > 0$ and $0 < \delta < 1$,

runs in polynomial time in the size of D, $1/\varepsilon$ and $\log(1/\delta)$, and

produces a random variable **Approximation**(D,ε,δ) such that

$$\Pr(|\mathbf{Approximation}(D,\varepsilon,\delta) - \mathrm{RF}(Q,D,\Sigma)| \leq \varepsilon \cdot \mathrm{RF}(Q,D,\Sigma)) \geq 1 - \delta$$

Sampling

fix a set of primary keys Σ, and a Boolean CQ Q

Sample[Σ,Q]

Input: a database D

Output: 0 or 1

$\{B_1, B_2, \ldots, B_n\}$ is a partition of D such that

each B_i collects conflicting (w.r.t. Σ) atoms of D

Repair := $\emptyset$

for i = 1 **to** n **do**

  choose $P(\mathbf{t}) \in B_i$ with probability $1/|B_i|$

  Repair := Repair $\cup$ $\{P(\mathbf{t})\}$

if Repair $\models$ Q **then**

  return 1

else

  return 0

end

Effgicient Approximation for RF-BCQA[Σ,Q](**CQ**)

fix a set of primary keys Σ, and a Boolean CQ Q

Approximation[Σ,Q]

Input: a database D, and numbers $\varepsilon > 0$ and $0 < \delta < 1$

Output: random number in [0,1]

Experiments := $((2+\varepsilon) \cdot m^k) / \varepsilon^2 \cdot \ln(2/\delta)$, where k is the number of atoms in Q, and m is the size of the largest B_i

Sum := ∅

Counter := ∅

repeat

  Sum := Sum + **Sample**[Σ,Q](D)

  Counter := Counter + 1

until Counter = Experiments

return Sum/Experiments

Effgicient Approximation for RF-BCQA[Σ,Q](**CQ**)

fix a set of primary keys Σ, and a Boolean CQ Q

Theorem: Approximation[Σ,Q] is an FPRAS for BCQA[Σ,Q](**CQ**)

Recap

- Inconsistent databases - do not conform with the integrity constraints coming with the underlying schema (such as primary keys)
- Consistent query answering (CQA) - find meaningful answers to queries when databases are inconsistent
- CQA is a hard problem, even in data complexity for CQs and primary keys
- Isolate classes of queries for which CQA is efficient in data complexity
- Provide data-efficient approximations schemes