Dynamic Complexity of Planar 3-connected Graph Isomorphism

Jenish C. Mehta
Dynamic Complexity

**Fixed Problem**

- Input

- *slight* change

- Computed Solution

Complexity of updating the solution?
Dynamic Complexity

Fixed Problem

Input
A Relation filled with tuples

slight change
Insertion/Deletion of a tuple

Computed Solution
A set of Relations

Complexity of updating the solution?
Complexity Class in which the Relations can be updated?
Definition. For any static complexity class $C$, we define its dynamic version, $\text{DynC}$ as follows: Let $\rho = \langle R_1^{a_1}, ..., R_s^{a_s}, c_1, ..., c_t \rangle$, be any vocabulary and $S \subseteq \text{STRUC}(\rho)$ be any problem. Let $R_{n,\rho} = \{\text{ins}(i, a'), \text{del}(i, a'), \text{set}(j, a) \mid 1 \leq i \leq s, \ a' \in \{0, ..., n - 1\}^{a_i}, 1 \leq j \leq t\}$ be the request to insert/delete tuple $a'$ into/from the relation $R_i$, or set constant $c_j$ to $a$. 

Let $\text{eval}_{n,\rho} : R_{n,\rho}^* \to \text{STRUC}(\rho)$ be the evaluation of a sequence or stream of requests. Define $S \in \text{DynC}$ iff there exists another problem $T \subseteq \text{STRUC}(\tau)$ (over some vocabulary $\tau$) such that $T \in \text{C}$ and there exist maps $f$ and $g$:

$$f : R_{n,\rho}^* \to \text{STRUC}(\tau), \ g : \text{STRUC}(\tau) \times R_{n,\rho} \to \text{STRUC}(\tau)$$

satisfying the following properties:

1. **(Correctness)** For all $r' \in R_{n,\rho}^*$, $(\text{eval}_{n,\rho}(r') \in S) \iff (f(r') \in T)$

2. **(Update)** For all $s \in R_{n,\rho}$, and $r' \in R_{n,\rho}^*$, $f(r's) = g(f(r'), s)$

3. **(Bounded Universe)** $\|f(r')\| = \|\text{eval}_{n,\rho}(r')\|^O(1)$

4. **(Initialization)** The functions $g$ and the initial structure $f(\emptyset)$ are computable in $C$ as functions of $n$. 
**Problem:** Vertex-colouring a graph using 3 colours?

**Input:** Relation (graph) $G(x,y)$
(a,b), (b,c), (c,d), (d,e), (e,f),
(a,c), (b,e), (b,f), (c,f), (g,h)

**Solution:**
$R = a,e,g$  $B = b,d,h$  $G = c,f,i$

**Change:** Insertion/Deletion of an edge, or tuple in $G$
Dynamic Complexity

**Problem:** Vertex-colouring a graph using 3 colours?

**Relations Maintained:**
A(x,y), B(x,y,z,w), R(p,q,r),
D(a,b,c,d,e), C(s,r)

**Dynamic Complexity:**
Complexity class C, to update the relations A,B,C,D,R and find the solution from them after insertion/deletion

Problem is in DynC
Dynamic Complexity

Problem: Parity of the String?

Input: Relation (string) $S(p,b)$

Relations:
$B(z) = \text{To find the parity of the string.}$
$\text{The only tuple in the relation will be the parity of the string.}$

Simple $DynP, DynL$ solution
Dynamic Complexity

**Problem:** Parity of the String?

**Input:** Relation (string) $S(p,b)$

**Relations:**
- $A(x,y) =$ To store the old string
- $B(z) =$ To find the parity of the string.

The only tuple in the relation will be the parity of the string.

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 1 | 0 | * | 0 | * | 1 |
Dynamic Complexity

Problem: Parity of the String?

\[ S(p,b) = (0,1), (1,0), (2,1), (3,1), (4,1), (5,0), (7,0), (9,1) \]

\[ A(x,y) = (0,1), (1,0), (2,1), (3,1), (4,1), (5,0), (7,0), (9,1) \]

\[ B(z) = (1) \]
Problem: Parity of the String?

User: insert(p,b)

A'(x,y) = A(x,y) OR
      x=p AND y=b

101110 * 0 * 1
0123456789
User: $insert(p, b)$
[assume insert(6, 1)]

$B'(z) = A(p, b) \land B(z) \lor \neg A(p, b) \land b = 0 \land B(z) \lor b = 1 \land z = 1 \land B(0) \lor z = 0 \land B(1)$
\[ R_2(v, x) = \text{BFS} \text{Edge}(v, a, b) \land \text{Path}(v, v, x, \{a, b\}) \]
\[ R_1(v, y) = \neg R_2(v, y) \]
\[ PR(v, s, t) = R_1(v, s) \land R_2(v, t) \land \text{Edge}(s, t) \quad \{\text{All edges connecting } R_1 \text{ and } R_2\} \]
\[ l_{\text{min}}(v, w) \leftarrow \min\{\text{level}_v(s) + 1 + \text{level}_t(w) : \ PR(v, s, t)\} \quad \{\text{Length of the new shortest path from } v \text{ to } w\} \]
\[ PR_{\text{min}}(v, w, s, t) = R_2(v, w) \land PR(v, s, t) \land (\text{level}_v(s) + 1 + \text{level}_t(w) = l_{\text{min}}(v, w)) \quad \{\text{Set of edges that lead to the shortest path}\} \]
\[ PR_{\text{lex}, \text{min}}(v, w, s, t) = PR_{\text{min}}(v, w, s, t) \land (s \leq t) \land (\forall p, q, PR_{\text{min}}(v, w, p, q) \Rightarrow (s < p) \lor ((s = p) \land (t \leq q))) \]
\{Choosing the lexicographically smallest edge. \( PR_{\text{lex}, \text{min}} \) is the set of new edges that will be added. The queries are now exactly similar to insertion of edges\}

\[ |P_2| < |P_1| \text{ or } |P_1| = |P_2| \land P_2 <_c P_1, \text{ and } \{x, y, z\} \text{ are on } |P_2| \]
\[ (l_{\text{old}} > l_{\text{new}}) \lor (l_{\text{old}} = l_{\text{new}} \land n_1 > n_2) \text{ and} \]
\[ \forall (C\text{Path}(v, v_e, v, \alpha, \{x, y, z\}) \land C\text{Path}(v, v_e, x, y, z)) \quad \{\text{All on the path from } v \text{ to } \alpha\} \]
\[ \forall (C\text{Path}(\beta, \beta_e, w, \{x, y, z\}) \land C\text{Path}(\beta, \beta_e, x, y, z)) \quad \{\text{All on the path from } \beta \text{ to } w\} \]
\[ \forall (C\text{Path}(v, v_e, v, \alpha, \{x\}) \land C\text{Path}(\beta, \beta_e, \beta, w, \{y, z\}) \land C\text{Path}(\beta, \beta_e, \beta, y, z)) \quad \{x \text{ on } \text{path}_{v, v_e}(v, \alpha) \text{ and } y, z \text{ on } \text{path}_{\beta, \beta_e}(\beta, w)\} \]
\[ \forall (C\text{Path}(v, v_e, v, \alpha, \{x, z\}) \land C\text{Path}(v, v_e, v, z, x) \land C\text{Path}(\beta, \beta_e, \beta, w, y)) \quad \{x, z \text{ on } \text{path}_{v, v_e}(v, \alpha) \text{ and } y \text{ on } \text{path}_{\beta, \beta_e}(\beta, w)\} \]
\{\text{EmbPar}(v, v_e, x, n_p) \text{ denotes that the embedding number of } x\text{'s parent in } [v, v_e] \text{ is } n_p\}
\[ \text{EmbPar}(v, v_e, x, n_p) = \exists x_p, \text{Parent}(v, v_e, x_p, x) \land \text{Emb}(x, x_p, n_p) \]
\[ \text{Emb}_p(v, v_e, t, x, n_x) = \text{Edge}(x, t) \land \exists n_p, d_t, n_{\text{old}}, \text{Deg}(t, d_t) \land \text{EmbPar}(v, v_e, t, n_p) \land \text{Emb}(t, x, n_{\text{old}}) \land (n_{\text{old}} \geq n_p \Rightarrow n_x = n_{\text{old}} - n_p) \land (n_{\text{old}} < n_p \Rightarrow n_x = n_{\text{old}} + d_x - n_p) \]
\[ \text{Emb}_f(v, x, n_x) = \exists n_{\text{old}}, d_v, \text{Emb}(v, x, n_{\text{old}}) \land \text{Deg}(v, d_v) \land (n_x = d_v - 1 - n_{\text{old}}) \]
\[ R_2(v, x) = BFSEdge(v, a, b) \land Path(v, x, \{a, b\}) \]
\[ R_1(v, y) = \neg R_2(v, y) \]
\[ PR(v, s, t) = R_1(v, s) \land R_2(v, t) \land Edge(s, t) \] \{All edges connecting \(R_1\) and \(R_2\)\}
\[ l_{\text{min}}(v, w) \leftarrow \min\{level_v(s) + 1 + level_t(w) : PR(v, s, t)\} \] \{Length of the new shortest path from \(v\) to \(w\)\}
\[ PR_{\text{min}}(v, w, s, t) = R_2(v, w) \land PR(v, s, t) \land (level_v(s) + 1 + level_t(w) = l_{\text{min}}(v, w)) \] \{Set of edges that lead to the shortest path\}
\[ PR_{\text{lex}, \text{min}}(v, w, s, t) = PR_{\text{min}}(v, w, s, t) \land (s \leq t) \land (\forall p, q. PR_{\text{min}}(v, w, p, q) \Rightarrow (s < p) \lor ((s = p) \land (t \leq q))) \] \{Choosing the lexicographically smallest edge. \(PR_{\text{lex}, \text{min}}\) is the set of new edges that will be added. The queries are now exactly similar to insertion of edges\}
\[ ||P_2|| < ||P_1|| \text{ or } ||P_1|| = ||P_2|| \land P_2 <_e P_1, \text{ and } \{x, y, z\} \text{ are on } ||P_2|| \]
\[ l_{\text{old}} > l_{\text{new}} \lor (l_{\text{old}} = l_{\text{new}} \land n_1 > n_2) \]
\[ (CPath(v, v_e, v, \alpha, \{x, y, z\}) \land CPath(v, v_e, x, y, z)) \] \{All on the path from \(v\) to \(\alpha\)\}
\[ \forall(CPath(\beta, \beta_e, w, \{x, y, z\}) \land CPath(\beta, \beta_e, x, y, z)) \] \{All on the path from \(\beta\) to \(w\)\}
\[ \forall(CPath(v, v_e, v, \alpha, \{x\}) \land CPath(\beta, \beta_e, 3, w, \{y, z\}) \land CPath(\beta, \beta_e, 3, w, y)) \] \{\(x\) on \(path_{\alpha,v_e}(v, \alpha)\) and \(y, z\) on \(path_{\beta,\beta_e}(\beta, w)\)\}
\[ \forall(CPath(v, v_e, v, \alpha, \{x\}) \land CPath(v, v_e, v, z, v) \land CPath(\beta, \beta_e, 3, w, y)) \] \{\(x, z\) on \(path_{\alpha,v_e}(v, \alpha)\) and \(y\) on \(path_{\beta,\beta_e}(\beta, w)\)\}
\{\(EmbPar(v, v_e, v, \alpha, n_p)\) denotes that the embedding number of \(v\)'s parent in \([v, v_e]\) is \(n_p\)\}
\[ EmbPar(v, v_e, v, x, n_p) = \exists x_p. Parent(v, v_e, x_p, x) \land Emb(x, x_p, n_p) \]
\[ Emb_p(v, v_e, t, x, n_x) = Edge(x, t) \land \exists n_p, d_t, n_{\text{old}}, Dcg(t, d_t) \land EmbPar(v, v_e, t, n_p) \land Emb(t, x, n_{\text{old}}) \land (n_{\text{old}} \geq n_p \Rightarrow n_x = n_{\text{old}} - n_p) \land (n_{\text{old}} < n_p \Rightarrow n_x = n_{\text{old}} + d_x - n_p) \]
\[ Emb_f(v, x, n_x) = \exists n_{\text{old}}, d_v, Emb(v, x, n_{\text{old}}) \land Dcg(v, d_v) \land (n_x = d_v - 1 - n_{\text{old}}) \]
Dynamic Complexity

Parity is NOT in $FO$ (uniform $AC^0$)

Parity is in $DynFO$!

Undirected Reachability is in $DynFO$!
Dynamic Complexity

DST (’93) – FOIES, Acyclic Reach
IP (’97) – Dynamic Complexity, Undirected Reach
Hesse (’01) – Reach in DynTC⁰
HI (’02) – Complete problems for DynC
DHK (‘14) – Triangulated PlanarReach in DynFO
Schwentick (‘13) – Perspectives
# Isomorphism in PlanarLand

|                        | Trees          | 3-connected planar graphs            | Planar Graphs               |
|------------------------|----------------|--------------------------------------|-----------------------------|
| **Quadratic/Linear time** | Elementary     | Weinberg (‘66); Hopcroft, Tarjan (‘73) | Hopcroft, Wong (‘74)        |
| **Logspace**            | Lindell (‘92)  | Datta, Limaye, Nimbhorkar (‘08)       | Datta, Limaye, Nimbhorkar, Thierauf, Wagner (‘09) |
| **DynFO**               | Etessami (‘98) | **This work**                         | ?                           |
This work

Main Results:

1. Breadth-First Search for general undirected graphs is in $DynFO$

2. Isomorphism for Planar 3-connected graphs is in $DynFO+$
Breadth-First Search in DynFO

(general undirected graphs)
Breadth-First Search in DynFO
(general undirected graphs)

Main Idea:
Maintain BFS-tree from every vertex in the graph
Breadth-First Search in DynFO
(general undirected graphs)

- **Edge** $(x, y)$
  - $(a, b), (b, a), (b, c), (c, b), ...$

- **Level** $(v, x, l)$
  - $(a, b, 1), (a, d, 2), ...$

- **BFSEdge** $(v, x, y)$
  - $(a, a, b), (a, b, e), ...$

- **Path** $(v, x, y, z)$
  - $(a, e, d, b), (a, a, d, c), ...$
Breadth-First Search in DynFO
(general undirected graphs)

Lemma 1:
After the insertion of edge \(\{a,b\}\), the level of a vertex \(x\) cannot change both in the BFS trees of \(a\) and \(b\).
Lemma 2: If any vertex $t$ lies on $\text{path}(b,b,w)$ and on $\text{path}(v,v,a)$, then the shortest path from $v$ to $x$ does not change after the insertion of $(a,b)$.
Breadth-First Search in DynFO
(general undirected graphs)

insert \((a,b)\)

- Find the shorter path:
  \(\text{path}(a,a,x)\) or \(\text{path}(b,b,x)\)
  [Lemma 1]

- Only New path to consider:
  \(\text{path}(v,v,a) + (a,b) + \text{path}(b,b,x)\)
Breadth-First Search in DynFO
(general undirected graphs)

**insert** (a,b)

- Find the shorter path:
  - \( \text{path}(v,v,x) \) or
  - \( \text{path}(v,v,a) + (a,b) + \text{path}(b,b,x) \)

  [Lemma 2]

- Update the relations if new path is shorter
Breadth-First Search in DynFO
(general undirected graphs)

**BFSEdge(v,x,y):**

Edge \((x,y)\) belongs to the BFS tree of vertex \(v\), if:
There exists a vertex \(w\) in BFS tree of \(v\) whose level has not changed AND \((x,y)\) lies on the path from \(v\) to \(w\) OR ...

![Graph diagram](image)
Breadth-First Search in DynFO (general undirected graphs)

BFSEdge(v,x,y):

... OR
There exists a vertex \( w \) in BFS tree of \( v \) whose level has changed AND \((x,y)\) lies on the path from \( v \) to \( a \) OR the path from \( b \) to \( w \) OR is \((a,b)\).
Breadth-First Search in DynFO
(general undirected graphs)

Path(v,x,y,z):
Breadth-First Search in DynFO
(general undirected graphs)

delete(a,b):
Breadth-First Search in DynFO
(general undirected graphs)

Lemma 3:
When an edge \((a, b)\) separates a set of vertices \(T\) from the BFS tree of \(v\), and \(r\) and \(x\) are vertices belonging to \(T\), then \(\text{path}(r, r, x)\) cannot pass through \((a, b)\)
Breadth-First Search in DynFO
(general undirected graphs)

Consistency?
A Theorem of Whitney

Theorem (Whitney, 1933):
A planar 3-connected graph has a unique embedding on the sphere

Anti/clockwise from $d$: 
$e \ b \ a \ f \ e$

*Impossible* to re-draw such that ordering is: 
$e \ a \ f \ b \ e$
Embedding a planar 3-connected graph

Emb \((v, x, n)\):
(d, a, 1),
(g, e, 3), ...

Face \((f, x, y, z)\):
(F, e, g, f),
(F, d, f, d),
(F, g, d, e), ...
Embedding a planar 3-connected graph

Lemma:
Two distinct vertices lie on at most one face in a 3-connected planar graph.
Canonical Breadth-First Search
(Thierauf, Wagner, 2007)
Canonical Breadth-First Search in DynFO+
(planar 3-connected graphs)

Key Idea:
Maintain CBFS-trees from every vertex, for every edge taken as the starting embedding edge
Canonical Breadth-First Search in DynFO+ (planar 3-connected graphs)

Edge \((x, y)\), Level \((v, x, l)\)

CBFSEdge \((v, q, x, y)\):
\((b, d, c, g),\)
\((b, d, b, a), \ldots\)

CPath \((v, q, x, y, z)\):
\((b, d, f, g, c),\)
\((b, d, e, f, d), \ldots\)
Canonical Breadth-First Search in DynFO+ (planar 3-connected graphs)

 Canonical Ordering on Paths:  $P_1 \prec_c P_2$ if

- $|P_1| < |P_2|$ OR
- $|P_1| = |P_2|$ AND
  $d = \text{lca}(x,y)$,
  $\text{emb}(d,dp) = 0$,
  $\text{emb}(d,dx) < \text{emb}(d,dy)$
Canonical Breadth-First Search in DynFO+ (planar 3-connected graphs)

insert (a,b)
Canonical Breadth-First Search in DynFO+ (planar 3-connected graphs)

\textbf{delete}(a,b):
Use $<_c$ relation to find the edge $(p,r)$
Canon from a CBFS tree

Canon(v,q,x) =
{
    (l, m) :
    for some ancestor w of x, let pw be the parent of w, ppw be the parent of pw, emb (v, q, pw, ppw) = 0, l = level (v, w) AND m = emb (v, q, pw, w)
}
Canon from a CBFS tree

Starting vertex: $b$
Starting edge: $(b,d)$

Pre-canon:
\[
\begin{align*}
(a) &= \{ (b,0), (a,2) \} \\
(b) &= \{ (b,0) \} \\
(c) &= \{ (b,0), (c,1) \} \\
(d) &= \{ (b,0), (d,0) \} \\
(e) &= \{ (b,0), (d,0), (e,3) \} \\
(f) &= \{ (b,0), (d,0), (f,2) \} \\
(g) &= \{ (b,0), (c,1), (g,2) \}
\end{align*}
\]
Canon from a CBFS tree

Starting vertex: $b$
Starting edge: $(b,d)$

Canon:
(a) = \{(0,0), (1,2)\}
(b) = \{(0,0)\}
(c) = \{(0,0), (1,1)\}
(d) = \{(0,0), (1,0)\}
(e) = \{(0,0), (1,0), (2,3)\}
(f) = \{(0,0), (1,0), (2,2)\}
(g) = \{(0,0), (1,1), (2,2)\}
Canon from a CBFS tree

Canon:
(a) = \{ (0,0), (1,2) \}
(b) = \{ (0,0) \}
(c) = \{ (0,0), (1,1) \}
(d) = \{ (0,0), (1,0) \}
(e) = \{ (0,0), (1,0), (2,3) \}
(f) = \{ (0,0), (1,0), (2,2) \}
(g) = \{ (0,0), (1,1), (2,2) \}

Canon for the Graph: \[ Canon(G,b,d) = \{ \{ (0,0), (1,2) \}, \{ (0,0), (1,0) \} \}, \ldots \} \]
Isomorphism

Testing for isomorphism between $G$ and $H$:
Graphs $G$ and $H$ are isomorphic if and only if:
For some starting vertex/edge pair $(v,q)$ in $G$,
There exists a vertex/edge pair $(w,r)$ in $H$,
Such that, $\text{Canon}(G,v,q) = \text{Canon}(H,w,r)$
Open Problems

Is Planar Graph Isomorphism decidable in $DynFO$?  
Yes

Does the dynamic version of every language in $L$ belong to $DynFO$?  
No

(Static Complexity) Upper Bound for $DynFO$?
Thank You