Combinatorial Optimization and Reasoning with Graph Neural Networks

Quentin Cappart, Didier Chetelat, Elias B. Khalil, Andrea Lodi,
Christopher Morris, Petar Velickovi

Amanda Salinas

Contenido

- Optimización combinatorial
- Uso ML en Optimización combinatorial
- Estado del arte en GNN for CO
- Conclusiones

Optimización Combinatorial

Optimización Combinatorial

- Campo interdisciplinario que abarca la optimización, la investigación de operaciones, las matemáticas discretas y la informática
- Problemas que implican optimizar una función de costo (u objetivo) al seleccionar un subconjunto de un conjunto finito, donde este último codifica las restricciones en el espacio de soluciones

Optimization Frameworks

- SAT (Satisfiability Problem):
  - Clausulas y literales en CNF
  - Aplicaciones: Hardware verification, planning
- CSP (Constraint Satisfaction Problem)
  - Variable, Dominios y restricciones
    - AllDifferent
  - Ejemplo: Sudoku
- ILP (Integer Linear Programming)
  - variables enteras, función objetivo y restricciones lineales
- MILP (Mixed-Integer Linear Programming)
  - variables enteras y no enteras, función objetivo y restricciones lineales
- PBO (Pseudo-Boolean Optimization)
  - variables booleanas, función objetivo y restricciones no lineales

TSP

Input: *A complete directed graph* G, *i.e.,* $E(G) = \{(u, v) \mid u, v \in V(G)\}$, *with edge costs* $w \colon E(G) \to \mathbb{R}$.
Output: *A permutation of the nodes* $\sigma \colon \{0, \ldots, n-1\} \to V$ *such that*

$$\sum_{i=0}^{n-1} w\big((\sigma(i), \sigma((i+1) \bmod n)\big)$$

is minimal over all permutations, where $n = |V|$.

TSP to ILP

$$x_{ij} = \begin{cases} 1 & \textit{if the cycle goes from city } i \textit{ to city } j, \\ 0 & \textit{otherwise,} \end{cases}$$

and let $w_{ij} > 0$ *be the cost or distance of traveling from city* i *to city* j, $i \neq j$. *Then, the TSP can be written as the following ILP:*[4]

$$\min \sum_{i=1}^{n} \sum_{j \neq i, j=1}^{n} w_{ij} x_{ij}$$

$$\text{subject to} \sum_{i=1, i \neq j}^{n} x_{ij} = 1 \qquad j \in [n],$$

$$\sum_{j=1, j \neq i}^{n} x_{ij} = 1 \qquad i \in [n],$$

$$\sum_{i \in Q} \sum_{j \notin Q} x_{ij} \geq 1 \qquad \forall Q \subsetneq [n], |Q| \geq 2.$$

Técnicas de resolución

- Métodos exactos
  - Branch and Bound en ILP
  - CDCL y propagación para SAT
- Métodos Búsqueda Local y metaheurísticas
  - Utiliza una solución candidata inicial y generación de vecindarios
  - Metaheurísticas para escapara de óptimo local (Tabu search, algoritmos genéticos, Core boosted en MaxSAT)
- Algoritmos de Aproximation: producir soluciones satisfacibles dentro de alguna región cerca del óptimo

ML en Optimización Combinatorial

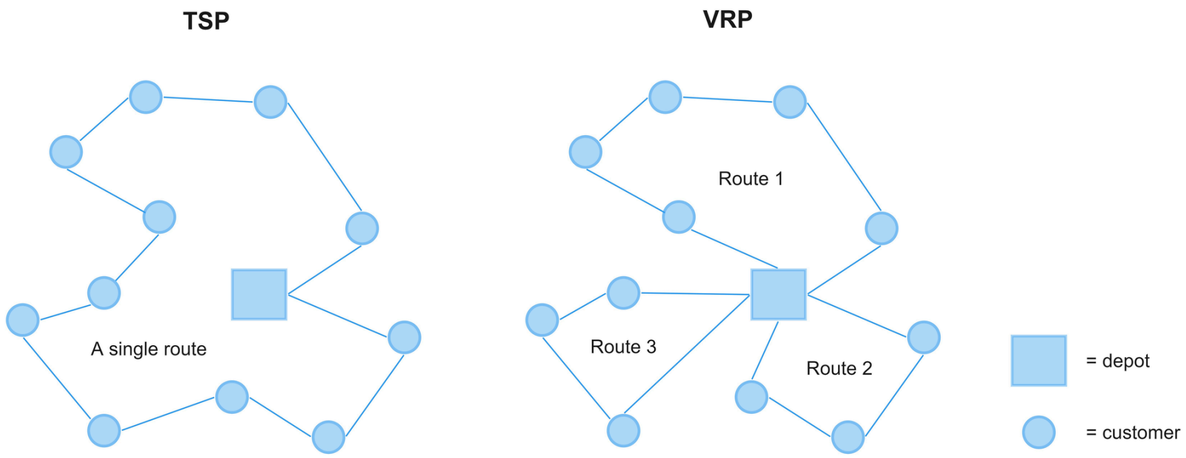
- Problemas OC son **NP-Hard**.
- Podemos aprender patrones?
  - VRP en una misma ciudad: Donde solo varían ligeramente los tiempos de viaje por condiciones de tráfico.
  - Problema de la Mochila: Pequeñas variaciones en el valor de los objetos pueden cambiar la satisfacibilidad de una solución.
- Distintas estrategias de uso:
  - Predecir Soluciones
  - Integrar a solver ya existente

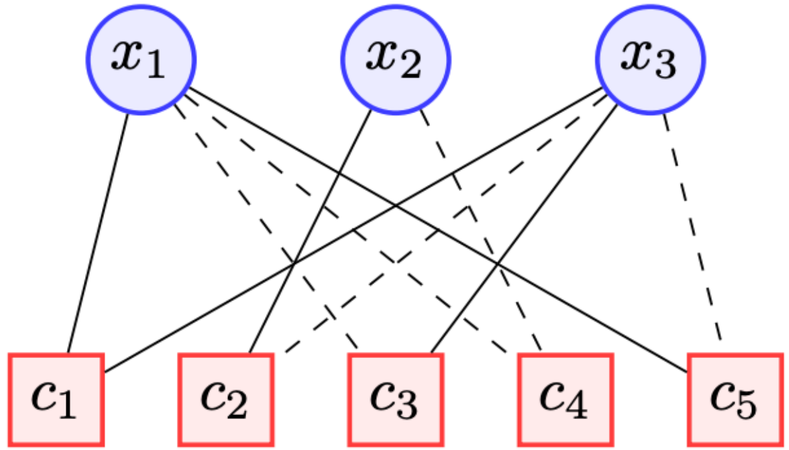
Desafíos en ML

- **Invariancia a Permutaciones:** Los métodos de ML deben ser robustos frente a cambios en el orden de los nodos en grafos.
- **Escalabilidad y Eficiencia en Datos Escasos**: Modelos eficientes en grafos grandes, capturando patrones con poca información etiquetada.
- **Capacidad para Integrar Información Extra**: Incorporar objetivos y restricciones adicionales en el modelo.
- **Generalización a Nuevas Instancias**: Capacidad para trabajar con instancias de diferentes tamaños y características.

GNN en Optimización Combinatoria

- Por qué GNN?
  - Estructura de grafo:
    - naturalmente como en TSP o VRP
    - Inducir un bipartito





GNN en Optimización Combinatoria

- Como resuelven desafíos anteriores
  - Invariancia: Las GNN explotan simetrías en los grafos, manteniendo invarianza a la permutación de nodos.
  - La naturaleza local de las GNN permite aprovechar la estructura de los grafos.
  - Integración de Información: Pueden manejar nodos y aristas con múltiples dimensiones, lo que facilita la integración de funciones objetivo y restricciones.
- Limitaciones
  - Escalabilidad: Aunque las GNN escalan linealmente con el número de aristas, los problemas de eficiencia en datos y complejidad aún están presentes.
  - Eficiencia de datos: La capacidad de eficiencia en datos sigue siendo un desafío, ya que los modelos supervisados requieren muchos datos para generalizar correctamente a problemas grandes o complejos.

GNN for CO
State of the art

Enfoques principales

- Enfoque primal: Encontrar buenas soluciones factibles
  - Predecir directamente soluciones de alta calidad o mejorar la velocidad de algoritmos heurísticos (warm-up)
  - Esto puede ser aprender:
- Enfoque dual: Certificar optimalidad o probar insatisfacibilidad

Enfoque primal

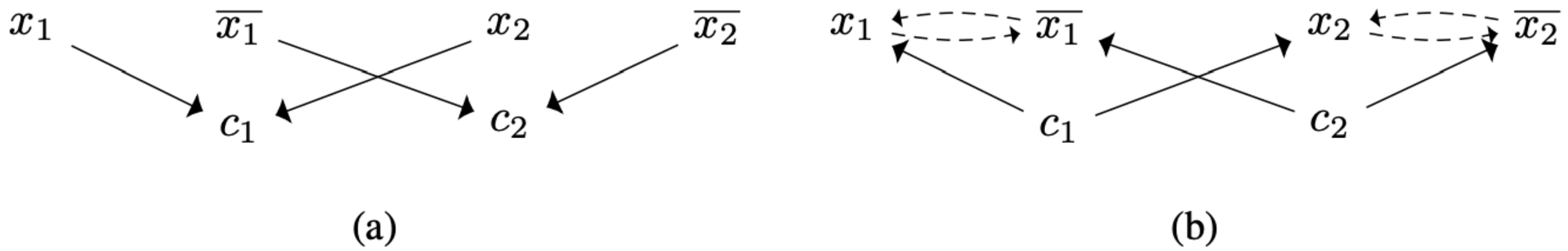
- Aprendizaje supervisado
- Aprendizae no supervisado
- Reinforcement

Aprendizaje supervisado

- Ejemplos:
  - Predicción de solución
  - Combinar en la búsqueda
- Limitaciones: datos etiquetados

Predicción de solución: NeuroSat

- Redes neuronales con paso de mensajes para predicir satisfacibilidad de un problema SAT
- Datos: a partir de ejecución de un SAT solver



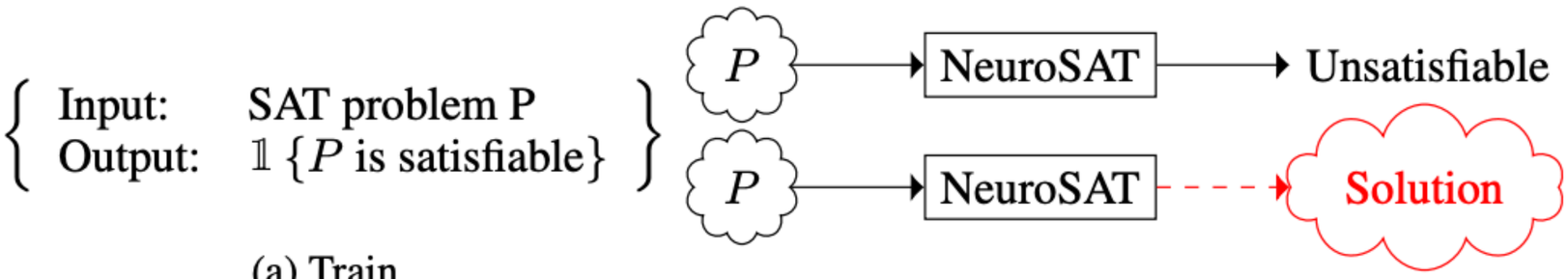
Predicción de solución: NeuroSat

$$(C^{(t+1)}, C_h^{(t+1)}) \leftarrow \mathbf{C}_{\text{u}}([C_h^{(t)}, M^\top \mathbf{L}_{\text{msg}}(L^{(t)})])$$
$$(L^{(t+1)}, L_h^{(t+1)}) \leftarrow \mathbf{L}_{\text{u}}([L_h^{(t)}, \text{Flip}(L^{(t)}), M\mathbf{C}_{\text{msg}}(C^{(t+1)})])$$

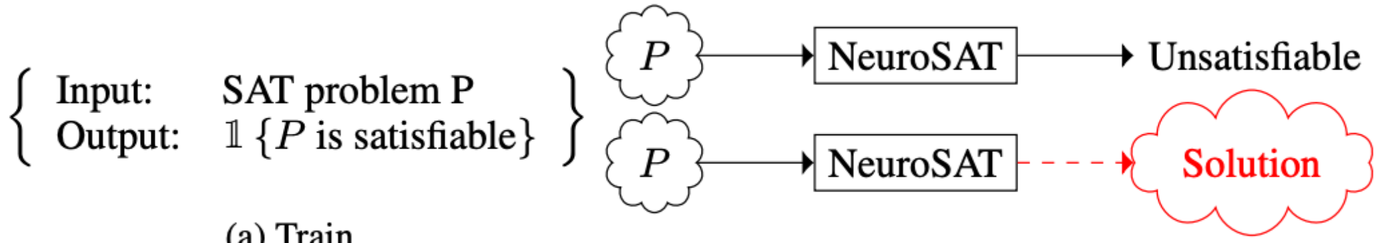
$$L_*^{(T)} \leftarrow \mathbf{L}_{\text{vote}}(L^{(T)}) \in \mathbb{R}^{2n},$$

$$y^{(T)} \leftarrow \text{mean}(L_*^{(T)})$$

Predicción de solución: NeuroSat

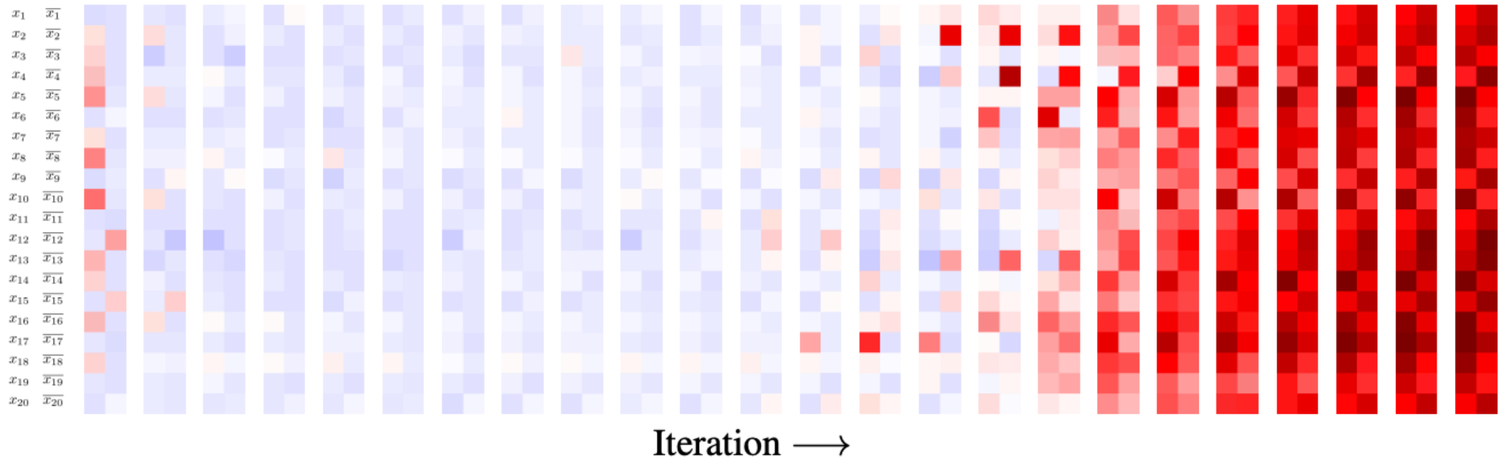


Predicción de solución: NeuroSat



(a) Train

(b) Test



Predicción de solución: NeuroSat

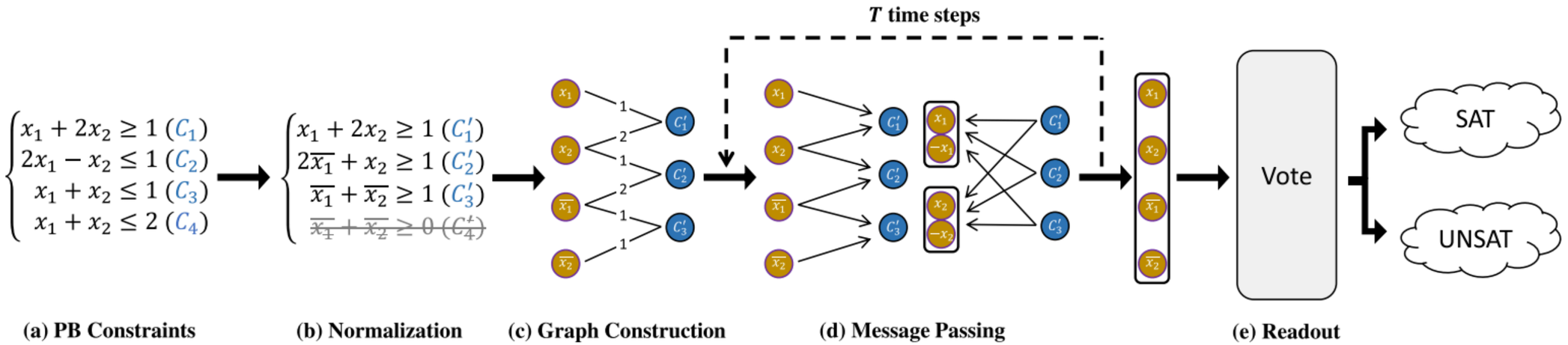
- El modelo se entrena con el conjunto SR(U(10, 40)), compuesto por problemas SAT aleatorios de 10 a 40 variables, generados de manera que uno sea satisfacible y otro insatisfacible al modificar un solo literal en una cláusula.
- Resultados: 85% accuracy en entrenamiento y 85% en problemas de otros dominios de test

Predicción de solución: NeuroSat

- Problemas con problemas insatisfacibles complejos: NeuroUNSAT
- Objetivo final: usar en optimización?

Predicción de solución: PBO

Learning the Satisfiability of Pseudo-Boolean Problem with Graph Neural Networks



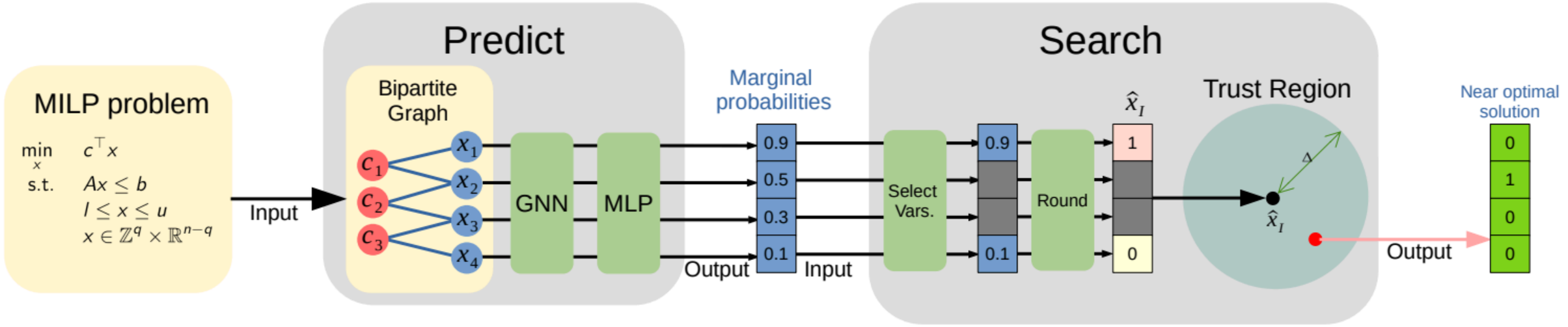
Predicción de solución: PBO

Learning the Satisfiability of Pseudo-Boolean Problem with Graph Neural Networks

Problem	#Variables			PB Cons.		Epochs	Accuracy	
	Train	*Valid*	*Test*	*#Cons.*	*Length*		*Valid*	*Test*
0–1KP	[3,10]	10	10	2.0	6.5	400	86.6%	86.1%
	[11,40]	40	40	2.0	25.5	400	87.3%	88.2%
	[41,100]	100	100	2.0	70.5	600	79.3%	79.5%
WIS	[3,10]	10	10	11.3	2.4	400	97.7%	97.9%
	[11,20]	20	20	59.3	2.2	400	92.1%	93.3%
	[21,30]	30	30	159.2	2.1	400	89.5%	88.9%
	[31,40]	40	40	309.3	2.1	600	86.0%	85.8%

Combinar en búsqueda: Predict and Search

- El objetivo es predecir probabilidades marginales de cada variable binaria en una instancia de MILP.
- En vez de fijar valores de manera rígida. Búsqueda local con región de confianza alrededor de una solución incial



Combinar en búsqueda: Predict and Search

- **Predict**

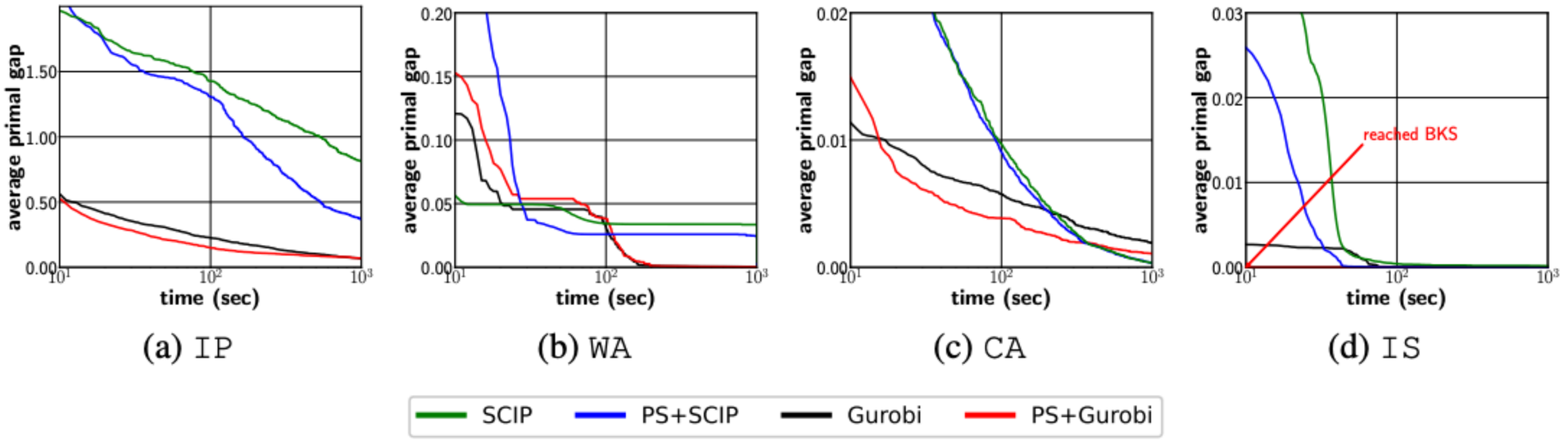
$$p(x;M) \equiv \frac{\exp(-E(x;M))}{\sum_{x'} \exp(-E(x';M))}, \quad \text{where } E(x;M) \equiv \begin{cases} c^\top x & \text{if } x \text{ is feasible,} \\ +\infty & \text{otherwise.} \end{cases}$$

$$L(\theta) \equiv -\sum_{i=1}^{N} \sum_{j=1}^{N_i} w^{i,j} \log P_\theta \left(x^{i,j}; M^i \right), \quad \text{where } w^{i,j} \equiv \frac{\exp\left(-{c^i}^\top x^{i,j}\right)}{\sum_{k=1}^{N_i} \exp\left(-{c^i}^\top x^{i,k}\right)}.$$

Combinar en búsqueda: Predict and Search

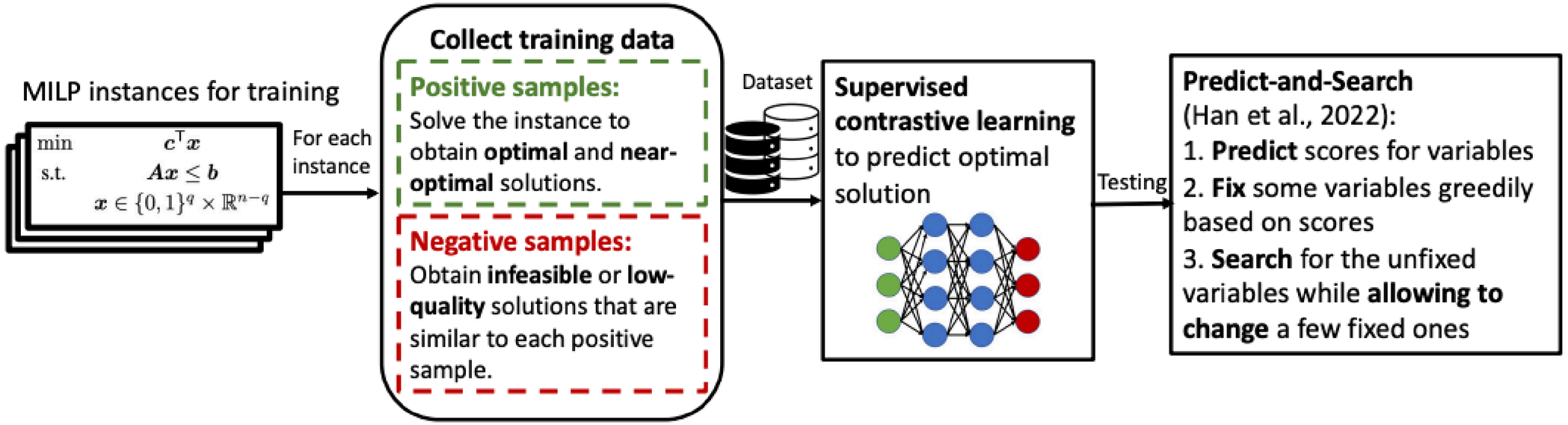
Datos y entrenamiento

- Cada dataset contiene 400 instancias: 240 train, 60 validation y 100 test
- Métrica de evaluación: relative primal gap entre los valores de la función objetivo



(a) IP (b) WA (c) CA (d) IS

Contrastive Predict and Search

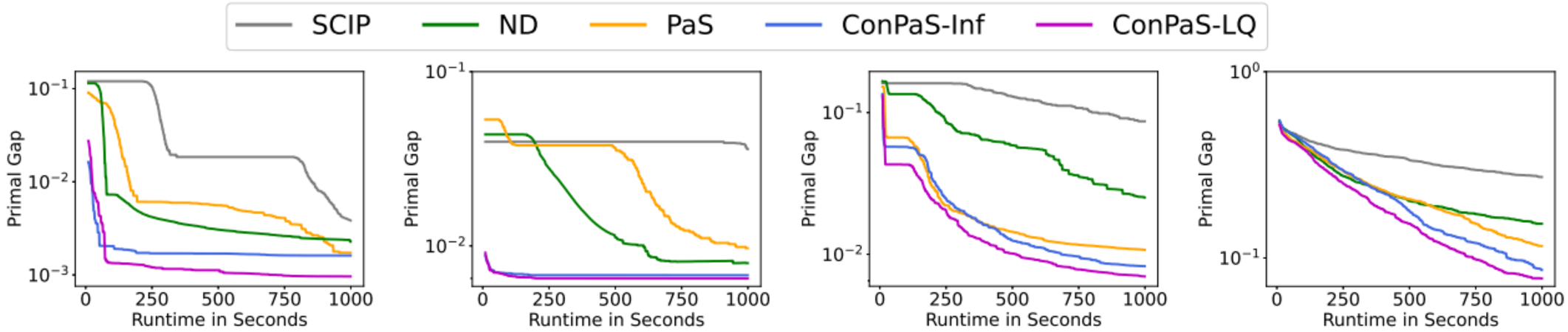


Contrastive Predict and Search

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{(\mathcal{S}_{\mathsf{p}}^{M}, \mathcal{S}_{\mathsf{n}}^{M}) \in \mathcal{D}} \frac{-1}{|\mathcal{S}_{\mathsf{p}}^{M}|} \sum_{\boldsymbol{x}_{\mathsf{p}} \in \mathcal{S}_{\mathsf{p}}^{M}} \log \frac{\exp(\boldsymbol{x}_{\mathsf{p}}^{\mathsf{T}} \boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}|M)/\tau(\boldsymbol{x}_{\mathsf{p}}|M))}{\sum_{\boldsymbol{x}' \in \mathcal{S}_{\mathsf{n}}^{M} \cup \{\boldsymbol{x}_{\mathsf{p}}\}} \exp(\boldsymbol{x}'^{\mathsf{T}} \boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}|M)/\tau(\boldsymbol{x}'|M))}$$

- M es conjunto de instancias de entrenamiento
- D es el conjunto de instancias positivas y negativas
- Función donde es bajo si el valor p(x|M) es similar a los positivos y disimilar a negativos
- Supervised Contrastive Loss

Contrastive Predict and Search



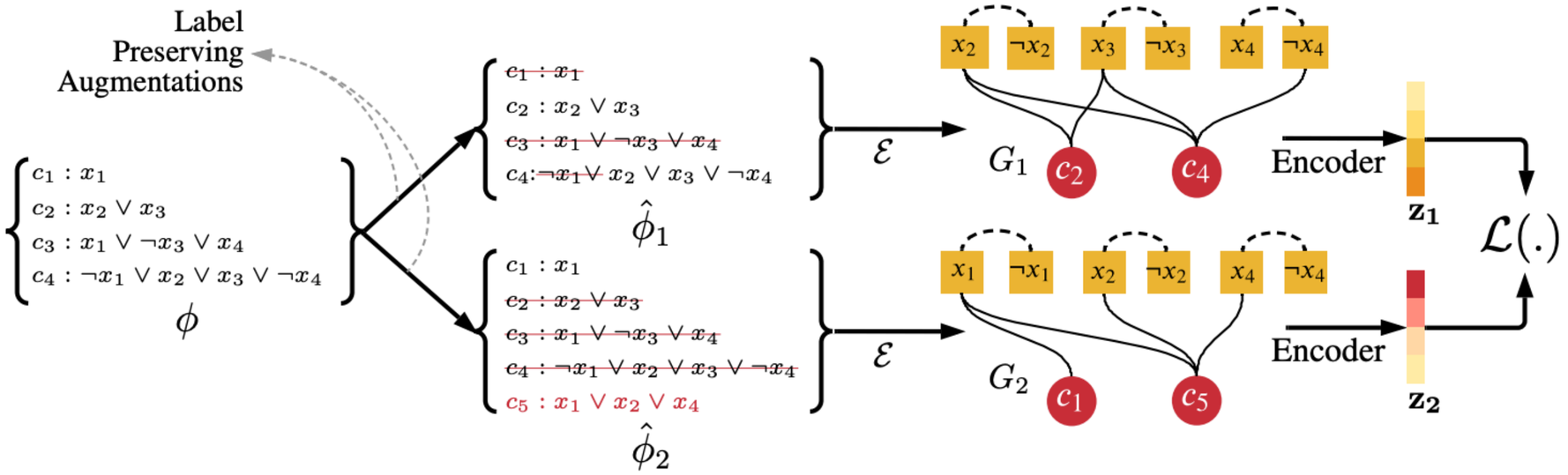
(a) MVC. (b) MIS. (c) CA. (d) IP.

Aprendizaje No supervisado

- Los enfoques en esta sección utilizan GNNs para producir soluciones que simultáneamente optimizan directamente el objetivo de un problema de optimización combinatoria y minimizan una medida de violación de restricciones
- Este método permite abordar instancias grandes sin necesidad de etiquetas, pero la calidad de la solución depende de un ajuste de la función de pérdida.

Aprendizaje No supervisado

NeuroSAT Contrastive



$$\mathcal{L}_{i,j} = -\log \frac{\exp(sim(\mathbf{m}_i, \mathbf{m}_j) \,/\, \tau)}{\sum_{k=1}^{2N} \mathbf{1}_{k \neq i} \exp(sim(\mathbf{m}_i, \mathbf{m}_k) \,/\, \tau)},$$

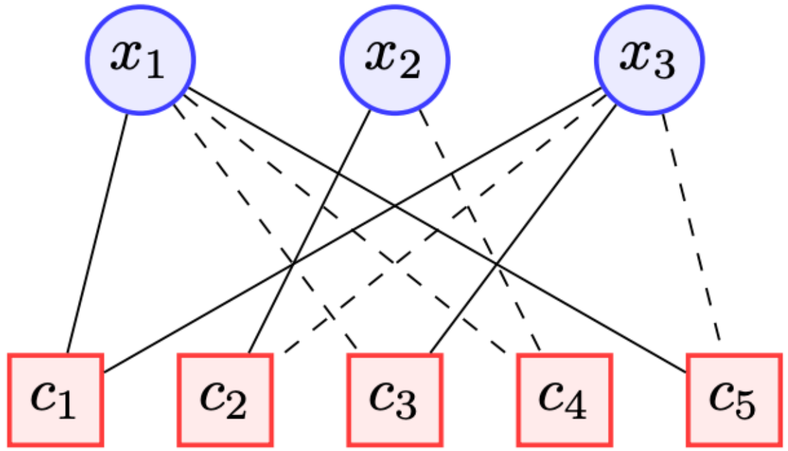
Reinforcement Learning

- Aprender heurísticas de búsqueda local para resolver SAT, minimizar la cantidad de pasos necesarios para encontrar asignaciones.

Algorithm 1 Local search for SAT

Input: Boolean formula ϕ, maximum number of trials K, maximum number of flips L

```
for i ← 1 to K do
    X ← Initialize(φ)
    for j ← 1 to L do
        if φ(X) = 1 then
            return X
        else
            index ← SelectVariable(φ, X)
            X ← Flip(X, index)
        end if
    end for
end for
return unsolved
```



Reinforcement Learning

- La GNN toma como entrada una fórmula booleana y una asignación de variables, y genera una probabilidad para cada variable

  - $M_{v+}^t : \mathbb{R}^{d_{t-1}} \to \mathbb{R}^{d_t}$ and $M_{v-}^t : \mathbb{R}^{d_{t-1}} \to \mathbb{R}^{d_t}$ compute the incoming messages to each variable from clauses they positively and negatively occur in.
  - $M_{c+}^t : \mathbb{R}^{d_{t-1}} \to \mathbb{R}^{d_t}$ and $M_{c-}^t : \mathbb{R}^{d_{t-1}} \to \mathbb{R}^{d_t}$ compute the incoming messages to each clause from variables that occur positively and negatively in them.
  - $U_v^t : \mathbb{R}^{d_{t-1}} \times \mathbb{R}^{d_t} \to \mathbb{R}^{d_t}$ and $U_c^t : \mathbb{R}^{d_{t-1}} \times \mathbb{R}^{d_t} \to \mathbb{R}^{d_t}$ update the representations of variables and clauses based on their previous representation and the sum of the incoming messages.
  - $Z : \mathbb{R}^{d_T} \to \mathbb{R}$ produces a score given the extracted node features of a variable.

Reinforcement Learning

- Experimentos: Evaluado frente a WalkSAT, demostrando menos pasos necesarios aunque con tiempo de ejecución más alto

	$\text{rand}_3(n, m)$	$\text{clique}_k(N, p)$	$\text{cover}_k(N, p)$	$\text{color}_k(N, p)$	$\text{domset}_k(N, p)$	WalkSAT
$\text{rand}_3(50, 213)$	**367** **273** **84%**	743 750 0%	749 750 0%	736 750 0%	642 750 20%	385 297 80%
$\text{clique}_3(20, 0.05)$ $n = 60, m = 1725$	529 750 48%	**116** **57** **100%**	623 750 16%	743 750 0%	725 750 0%	237 182 100%
$\text{cover}_5(9, 0.5)$ $n = 55, m = 790$	749 750 0%	750 750 0%	**181** **115** **100%**	750 750 0%	224 162 100%	319 280 96%
$\text{color}_5(20, 0.5)$ $n = 100, m = 480$	675 750 16%	748 750 0%	750 750 0%	**342** **223** **88%**	645 750 16%	416 379 80%
$\text{domset}_4(12, 0.2)$ $n = 60, m = 995$	729 750 0%	660 750 16%	304 169 76%	748 750 0%	**205** **121** **100%**	217 140 100%

Enfoque dual

- Certificar optimalidad: demostrar que es óptimo o cerca de un bound.
- Ejemplo
  - Integración de GNN en Branch and Bound
    - Branching variable de selección
    - Predicción de cortes en Branch and Cut
  - VSIDS en CDCL (SAT): similar a branching de variables

Algorithmic Reasoning

- Este enfoque va más allá de la simple predicción de soluciones o la optimización de la búsqueda, ya que se enfoca en entrenar modelos de GNN para aprender comportamientos algorítmicos o emular pasos de algoritmos en la resolución de problemas de CO.
- Un algoritmo se dividie en partes, donde cada una se pueda modelar a través de una red neuronal

Conclusiones

Conclusiones

- Limitaciones actuales
  - Expresividad
  - Generalización
  - Limitación de datos

Conclusiones

- Investigación futura
  - Entender cuando GNN aceleran solvers
  - Construir frameworks genericos