

روشی جدید برای حل مسائل ارضای محدودیت

غلامرضا قاسم‌ثانی* و مجید نمازی**
دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف
دانشگاه آزاد اسلامی (واحد خوی)

(دریافت مقاله: ۸۰/۱۱/۳ - دریافت نسخه نهایی: ۸۲/۱۱/۲۵)

چکیده - بسیاری از مسائل مطرح در زمینه هوش مصنوعی را می‌توان به صورت مسائل ارضای محدودیت^۱ توصیف کرد. این مسائل با استفاده از مجموعه‌ای از متغیرها و تعدادی محدودیت بر روی مقادیری که این متغیرها می‌توانند اختیار کنند، تعریف می‌شوند (در این نوع از مسائل از واژه "برچسب" نیز برای اشاره به "مقدار" یک متغیر استفاده می‌شود و لذا به آنها مسائل برچسب دهی سازگار^۲ نیز اطلاق می‌شود). حل این مسائل مجموعه‌ای از مقادیر منحصر به فرد برای متغیرهاست، به طوری که تمامی محدودیت‌های مورد نظر مسئله ارضا شده باشد. تا به حال تعدادی الگوریتم جستجو، ویژه حل این نوع از مسائل ارائه شده است که برخی از آنها با آینده‌نگری که در حین حل مسئله انجام می‌دهند، تعداد عقب‌گردهای^۳ کمتری انجام داده و در تعداد قدمهای کمتری به راه حل دست می‌یابند. این الگوریتمها عبارت‌اند از "بررسی جلورو"، "آینده‌نگر جزئی" و "آینده‌نگر کامل". این الگوریتمها از نظر میزان تلاشی که در هر مرحله در قالب بررسیهای سازگاری^۴، صرف آینده‌نگری می‌کنند و تعداد عقب‌گردهایی که در حین حل مسئله انجام می‌دهند، با یکدیگر تفاوت دارند. در این مقاله، ضمن تشریح الگوریتمهای ذکر شده، روش جستجوی جدیدی که آن را "آینده‌نگر کامل بهبود یافته" نامیده‌ایم نیز معرفی می‌شود که از الگوریتم آینده‌نگر کامل کارا تر است.

واژگان کلیدی: هوش مصنوعی، جستجو، مسائل ارضای محدودیت، مسائل برچسب دهی سازگار

A New Method for Solving Constraint Satisfaction Problems

G. Ghassem- Sani and M. Namazi

Assistant Professor Department of Computer Engineering, Shsrif University of Technology
Instructor, Islamic Azad University (Khoy Branch)

Abstract: Many important problems in Artificial Intelligence can be defined as Constraint Satisfaction Problems (CSP). These types of problems are defined by a limited set of variables, each having a limited domain and a number of Constraints on the values of those variables (these problems are also called Consistent Labeling Problems (CLP), in which "Labeling" means assigning a value to a variable.) Solution to these problems is a set of unique values for variables such that all the problem constraints are satisfied. Several search algorithms have been proposed for solving these problems, some of which reduce the need for backtracking by doing some sort of looking to future, and produce more efficient solutions. These are the so-called Forward Checking (FC), Partially Lookahead (PL), and Fully Lookahead (FL) algorithms. They are different in terms of the amount of looking to the future, number of backtracks that are performed, and the quality of the solution that they find. In this paper, we propose a new search algorithm we call Modified Fully Lookahead (MFL) which is Shown to be more efficient than the original Fully Lookahead algorithm

Keywords: Artificial Intelligence, Search, Constraint Satisfaction Problems, consistent labeling problems

** - مری

* - استادیار

۲) تعداد محدودی "محدودیت" بر روی مقادیری که این متغیرها می‌توانند به خود بگیرند.

حل این دسته از مسائل هوش مصنوعی نیز با کمک عمل جستجو انجام می‌شود. فضای جستجوی^{۱۲} این مسائل شامل وضعیتهایی است که در آنها مقدار تعدادی از متغیرهای مسئله تعیین شده است. وضعیت اولیه^{۱۳} صرفاً شامل تعریف مسئله است و در آن مقدار هیچ‌یک از متغیرها مشخص نیست. در وضعیت هدف^{۱۴} و یا وضعیتهای هدف (اگر مسئله بیشتر از یک جواب داشته باشد) مقدار کلیه متغیرها، با رعایت کلیه محدودیت‌های مسئله تعیین شده است [۱۵]. معمای حروف، هشت وزیر و رنگ کردن نقشه نمونه‌های نوعی از این دسته از مسائل هوش مصنوعی‌اند.

۳- الگوریتم‌های جستجوی آینده‌نگر ویژه حل مسائل ارضای محدودیت

مسائل ارضای محدودیت را می‌توان توسط روش‌های جستجوی عمومی سیستماتیک نظیر روش‌های "عمق اول"^{۱۵}، "سطح اول"^{۱۶}، "عمق اول با تعمیق مکرر"^{۱۷} حل کرد [۴] و [۸-۱۱]. لیکن تعدادی الگوریتم جستجو ویژه حل این دسته از مسائل هوش مصنوعی نیز ابداع شده است که کارایی بیشتری نسبت به روش‌های مذکور دارند. تعدادی از این روش‌ها که به آنها روش‌های آینده‌نگر می‌گوییم، با بررسی‌های بیشتری (نسبت به روش‌های عمومی) که در هنگام تعیین مقدار هر متغیر انجام می‌دهند، از عقبگردهای آتی جلوگیری کرده و در تعداد قدم‌های کمتری به راه حل مسئله دست می‌یابند. حتی تحت شرایط خاصی می‌توان برخی از مسائل ارضای محدودیت را حتی بدون عقبگرد حل کرد [۶] و [۱۳].

الگوریتم‌های جستجوی آینده‌نگری که تا به حال برای حل مسائل ارضای محدودیت ابداع شده‌اند، شامل: بررسی جلورو^{۱۸}، آینده‌نگر جزیی^{۱۹} و آینده‌نگر کامل^{۲۰} است [۷]. الگوریتم جدیدی نیز در این مقاله معرفی می‌شود که آن را "آینده‌نگر کامل بهبود یافته"^{۲۱} نامیده‌ایم. از این پس برای سادگی "بررسی

مسائل ارضای محدودیت بخش نسبتاً وسیعی از مسائل مطرح در زمینه هوش مصنوعی را در برمی‌گیرند. معمای حروف^۵، هشت وزیر^۶، رنگ کردن نقشه^۷ و درک تصویر توسط الگوریتم برچسب‌دهی والتز^۸، همگی مثالهایی از این نوع از مسائل اند [۱۴-۱۶ و ۱۸].

مؤلفه‌های اصلی مسائل ارضای محدودیت، متغیرها و محدودیت‌ها هستند. هدف نهایی در حل این‌گونه مسائل، مقداردهی به متغیرهاست، به گونه‌ای که مجموعه محدودیت‌های مسئله ارضا شوند. مقدار هر متغیر (در این نوع از مسائل از واژه "برچسب" نیز برای اشاره به "مقدار" یک متغیر استفاده می‌شود)، از دامنه مقادیر مجاز آن انتخاب می‌شود. انتخاب مقدار متغیرها بایستی به گونه‌ای صورت گیرد که مقادیر متغیرهای مختلف طبق محدودیت‌های تعریف شده، با یکدیگر سازگار^۹ باشند. دامنه مقادیر متغیرها توسط محدودیت‌های یکتایی^۱، و شرایط سازگاری مقادیر متغیرهای مختلف نسبت به یکدیگر اغلب با استفاده از محدودیت‌های چندتایی^{۱۱} (اغلب به صورت دوتایی) توصیف می‌شوند.

در ادامه ابتدا مسائل ارضای محدودیت تعریف می‌گردد. سپس در بخش (۳)، در مورد روش‌های جستجوی آینده‌نگر موجود ویژه حل این نوع از مسائل توضیح داده می‌شود. در بخش (۴)، الگوریتم جدید "آینده‌نگر کامل بهبود یافته" معرفی می‌شود. بخش (۵) به مقایسه الگوریتم جدید با الگوریتم‌های موجود و ارائه نتایج تجربی به دست آمده اختصاص دارد. در انتها در بخش (۶)، نتیجه‌گیری کار انجام شده ارائه می‌شود.

۲- مسائل ارضای محدودیت

یک مسئله ارضای محدودیت، یکی از انواع مسائل هوش مصنوعی است که با کمک دو مؤلفه اصلی زیر تعریف می‌شود [۱۷]:

۱) تعداد محدودی "متغیر" (که هر یک دارای یک دامنه محدود غیرتهی است) و

۳-۱- الگوریتم بررسی جلورو

در الگوریتم بررسی جلورو، مقادیری از دامنه متغیرهای آتی که با مقدار انتخاب شده برای متغیر جاری ناسازگارند، حذف می‌شوند؛ زیرا واضح است که این مقادیر در آینده نمی‌توانند برای آن متغیرهای آتی انتخاب شوند. در صورتی که دامنه یکی از متغیرهای آتی تهی شود، بایستی عقبگرد صورت گرفته و برای متغیر جاری و یا متغیرهای گذشته، مقدار دیگری انتخاب شود؛ در غیر این صورت جستجو با تعیین مقدار متغیر بعدی ادامه پیدا می‌کند. برای مثال، شکل (۱) مراحل اجرای الگوریتم بررسی جلورو را برای حل مسئله چهار وزیر نشان می‌دهد. (به منظور سادگی در توضیح الگوریتمها، از مسئله چهار وزیر که فضای جستجوی کوچکتری نسبت به مسئله هشت وزیر دارد، استفاده می‌شود.) همان‌گونه که در شکل مشاهده می‌شود، در مرحله هشتم، اولین جواب مسئله به دست می‌آید. در این شکل مقادیری از دامنه متغیرها که توسط الگوریتم بررسی جلورو حذف شده‌اند، با علامت FC و مواردی که در اثر عقبگرد حذف گردیده‌اند، با علامت BT مشخص شده است. علامت $\sqrt{\quad}$ نیز در شکل، مشخص کننده مقدار انتخاب شده برای هر متغیر است. الگوریتم FC به صورت شبه کد در بخش پیوست این مقاله آورده شده است.

توجه شود که در الگوریتم FC به محض آن‌که دامنه یکی از متغیرهای آتی تهی شود، بررسی جلورو متوقف شده و عقبگرد آغاز می‌شود. به عنوان مثال در مرحله دوم به دلیل آن‌که دامنه Q3 تهی می‌شود، بررسی جلورو متوقف شده و عقبگرد آغاز می‌گردد. لذا مقادیر دامنه Q4 در این مرحله دیگر مورد بررسی قرار نمی‌گیرد. به همین خاطر، همان‌گونه که در شکل ۱ نشان داده شده، مقدار ۳ از دامنه‌ی این متغیر، علیرغم ناسازگار بودن با مقدار متغیر جاری Q2، حذف نشده است. اما اگر فرضاً دامنه متغیر Q3 تهی نشده بود و پروسه عقبگرد آغاز نمی‌شد، بررسی جلورو ادامه می‌یافت و مقدار ۳ از دامنه Q4 نیز حذف می‌گردید.

جلورو" را با FC، "آینده نگر جزئی" را با PL، "آینده نگر کامل" را با FL، و "آینده نگر کامل بهبود یافته" را با MFL نشان می‌دهیم. در هر مرحله از حل مسئله مقدار یکی از متغیرها تعیین می‌شود که به آن "متغیر جاری" می‌گوییم؛ به متغیرهایی که در مراحل قبلی مقدارشان تعیین شده، متغیرهای گذشته و به آنهایی که در ادامه جستجو و پس از متغیر جاری مقدار خواهند گرفت، "متغیرهای آتی" می‌گوییم. ضمناً الگوریتمهای آینده‌نگر برای آن دسته از مسائل ارضای محدودیت قابل استفاده‌اند که دارای شرایط زیر باشند:

- ۱) دامنه هر متغیر، محدود و شامل مقادیر گسسته باشد،
- ۲) محدودیتهای مسئله تنها شامل محدودیتهای دودویی^{۲۲} باشد،
- ۳) محدودیتهای دودویی تعریف شده بین دو متغیر نسبت به هم قرینه باشد؛ بدین معنا که سازگاری مقادیر موجود در دامنه دو متغیر مختلف، مستقل از ترتیب مقدار دهی به هر یک از آنها باشد [۵].

به‌طور کلی الگوریتمهای جستجوی آینده‌نگر در ابتدای هر مرحله، مقداری برای متغیر جاری انتخاب می‌کنند. اگر این امر را به عنوان اضافه کردن محدودیتی جدید به مسئله در نظر بگیریم، الگوریتمهای جستجوی آینده‌نگر با کمک انتشار این محدودیت و با توجه به مقدار انتخاب شده برای متغیر جاری، دامنه متغیرهای آتی (آنهایی که هنوز مقداری برایشان تعیین نشده است) را با حذف برخی از مقادیر موجود در آنها، هرس^{۲۳} می‌کنند. میزان هرس در الگوریتمهای جستجوی آینده‌نگر مختلف، متفاوت است. اگر در اثر این عمل حذف رو به آینده، دامنه یکی از متغیرهای آتی تهی شود، عمل عقبگرد صورت گرفته و برای متغیر جاری و یا متغیرهای گذشته مقدار دیگری انتخاب می‌شود. در صورت عقبگرد دامنه متغیرها به وضعیت پیش از انتخاب نامناسب مقدار برای متغیرها اعاده می‌شود. در غیر این صورت جستجو با انتخاب مقداری برای متغیر بعدی ادامه پیدا می‌کند. در ادامه توضیح مختصری درباره هریک از این الگوریتمهای جستجوی آینده‌نگر ارائه می‌شود.

	۱	۲	۳	۴		۱	۲	۳	۴		۱	۲	۳	۴		۱	۲	۳	۴	
Q1	✓				(۱)	Q1	✓			(۲)	Q1	✓			(۳)	Q1	✓			(۴)
Q2	FC	FC				Q2	FC	FC	✓		Q2	FC	FC	BT	✓	Q2	FC	FC	BT	✓
Q3	FC		FC			Q3	FC	FC	FC	FC	Q3	FC		FC	FC	Q3	FC	✓	FC	FC
Q4	FC			FC		Q4	FC			FC	Q4	FC	FC		FC	Q4	FC	FC	FC	FC

	۱	۲	۳	۴		۱	۲	۳	۴		۱	۲	۳	۴		۱	۲	۳	۴	
Q1	BT	✓			(۵)	Q1	BT	✓		(۶)	Q1	BT	✓		(۷)	Q1	BT	✓		(۸)
Q2	FC	FC	FC			Q2	FC	FC	FC	✓	Q2	FC	FC	FC	✓	Q2	FC	FC	FC	✓
Q3		FC		FC		Q3		FC	FC	FC	Q3	✓	FC	FC	FC	Q3	✓	FC	FC	FC
Q4		FC				Q4		FC		FC	Q4	FC	FC		FC	Q4	FC	FC	✓	FC

شکل ۱- مراحل اجرای الگوریتم بررسی جلورو (FC) در حل مسئله چهار وزیر

مشاهده می‌شود که الگوریتم آینده‌نگر جزئی در طی مراحل کمتری (نسبت به الگوریتم بررسی جلورو) به حل مسئله رسیده است. علت این امر آن است که در الگوریتم آینده‌نگر جزئی، با کمک آینده‌نگری بیشتر، محدودیت بیشتری بر روی انتخابهای موجود برای متغیرهای آتی اعمال شده است. الگوریتم PL به صورت شبه کد در مرجع [۷] ارائه شده است.

۳-۳- الگوریتم آینده‌نگر کامل

الگوریتم آینده‌نگر کامل می‌تواند برای اعمال محدودیت بیشتری نسبت به الگوریتم آینده‌نگر جزئی مورد استفاده قرار گیرد. اختلاف الگوریتم آینده‌نگر کامل با الگوریتم آینده‌نگر جزئی در این است که در هر مرحله از الگوریتم آینده‌نگر جزئی، مقادیر موجود در دامنه هر یک از متغیرهای آتی تنها با دامنه متغیرهای آتی پس از آن بررسی می‌شود؛ در حالی که در هر مرحله از الگوریتم آینده‌نگر کامل، مقادیر هر یک از متغیرهای آتی با دامنه کلیه متغیرهای آتی دیگر (قبل و بعد از خود) بررسی می‌شود. شکل (۳) مراحل اجرای الگوریتم آینده‌نگر کامل را در حل مسئله چهار وزیر نشان می‌دهد. در این روش در مرحله پنجم، اولین جواب مسئله به دست می‌آید. در شکل، برای مشخص کردن مقادیری که به خاطر بررسی بیشتر (نسبت

۲-۳- الگوریتم آینده‌نگر جزئی

در الگوریتم جستجوی آینده‌نگر جزئی، در هر مرحله ابتدا (همانند روش بررسی جلورو) سازگاری مقدار متغیر جاری با مقادیر موجود در دامنه متغیرهای آتی بررسی شده و مقادیر ناسازگار با آن از دامنه متغیرهای آتی حذف می‌شود. سپس دامنه متغیرهای آتی نیز با یکدیگر بررسی می‌شوند و اگر مقداری از دامنه یک متغیر آتی، مثلاً V ، با تمامی مقادیر باقیمانده در دامنه یکی از متغیرهای آتی پس از آن ناسازگار باشد، آن مقدار نیز از دامنه V حذف می‌شود. در صورتی که دامنه یکی از متغیرهای آتی تهی شود، عمل عقبگرد صورت گرفته و برای متغیر جاری و یا متغیرهای گذشته، مقدار دیگری انتخاب می‌شود. شکل (۲) مراحل اجرای الگوریتم آینده‌نگر جزئی را در حل مسئله چهار وزیر نشان می‌دهد. همان‌گونه که مشاهده می‌شود، در مرحله ششم اولین جواب مسئله به دست می‌آید. در این شکل، مقادیری که در بررسی مقدار متغیر جاری با متغیرهای آتی حذف شده است با علامت FC، و مقادیری که در بررسی مقادیر متغیرهای آتی با یکدیگر حذف شده‌اند با علامت PL مشخص شده است.

با مقایسه مراحل مربوط به الگوریتم بررسی جلورو، شکل (۱) و مراحل مربوط به الگوریتم آینده‌نگر جزئی شکل (۲)

	۱	۲	۳	۴		۱	۲	۳	۴		۱	۲	۳	۴
Q1	✓					✓					BT	✓		
Q2	FC	FC	PL			FC	FC	PL	✓		FC	FC	FC	
Q3	FC	PL	FC			FC	PL	FC	FC			FC		FC
Q4	FC			FC		FC			FC			FC		

(۱) (۲) (۳)

	۱	۲	۳	۴		۱	۲	۳	۴		۱	۲	۳	۴
Q1	BT	✓				BT	✓				BT	✓		
Q2	FC	FC	FC	✓		FC	FC	FC	✓		FC	FC	FC	✓
Q3		FC	FC	FC		✓	FC	FC	FC		✓	FC	FC	FC
Q4		FC		FC		FC	FC		FC		FC	FC	✓	FC

(۴) (۵) (۶)

شکل ۲- مراحل اجرای الگوریتم آینده‌نگر جزئی (PL) در حل مسئله چهار وزیر

	۱	۲	۳	۴		۱	۲	۳	۴		۱	۲	۳	۴
Q1	✓					BT	✓				BT	✓		
Q2	FC	FC	PL			FC	FC	FC			FC	FC	FC	✓
Q3	FC	PL	FC	FL			FC	FL	FC		✓	FC	FL	FC
Q4	FC			FC		FL	FC		FL		FL	FC	✓	FL

(۱) (۲) (۳) (۴) (۵)

شکل ۳- مراحل اجرای الگوریتم آینده‌نگر کامل (FL) در حل مسئله چهار وزیر

الگوریتم دیگر نیست. به عبارت دیگر مزیت اصلی آن تعداد مراحل و عقبگرد کمتر است.

۴- الگوریتم آینده‌نگر کامل بهبود یافته

الگوریتم MFL با هدف اصلی افزایش کارایی الگوریتم آینده‌نگر کامل، از طریق کاهش تعداد بررسیهای سازگاری انجام شده در الگوریتم FL (و نه کاهش تعداد مراحل حل مسئله)، طراحی شده است [۱ و ۲]. به عبارت دیگر، همان‌گونه که در ادامه خواهیم دید، الگوریتم MFL می‌تواند مسائل ارضای محدودیت را با صرف انرژی کمتری از نظر تعداد بررسی سازگاری انجام شده (در مقایسه با روش FL)، و در عین حال با کیفیتی حداقل به خوبی روش FL (از نظر تعداد مراحل) حل کند. در این الگوریتم، همانند روش آینده‌نگر جزئی، مقادیر دامنه هریک از متغیرهای آتی تنها با دامنه متغیرهای آتی پس از

به روشهای قبلی) این روش حذف شده‌اند، از علامت FL استفاده شده است. همان‌گونه در شکل (۳) دیده می‌شود، الگوریتم آینده‌نگر کامل در طی مراحل کمتری (نسبت به الگوریتم آینده‌نگر جزئی) به اولین حل مسئله دست یافته است و این به علت اعمال محدودیت بیشتر بر روی انتخابهای متغیرهای آتی است. برای دیدن الگوریتم FL به صورت شبه کد به مرجع [۷] رجوع شود.

همان‌گونه که توضیح داده شد، الگوریتم آینده‌نگر کامل به نسبت از دو الگوریتم آینده‌نگر جزئی و بررسی جلورو، تعداد بررسیهای سازگاری بیشتری انجام داده و مسائل را در طی مراحل کمتری حل می‌کند. از طرفی تعداد عقبگردهای انجام شده نیز در این روش کمتر است. البته بایستی توجه شود که زمان کل حل مسئله در الگوریتم آینده‌نگر کامل به علت بررسیهای سازگاری بیشتری که انجام می‌دهد، لزوماً کمتر از دو

آن بررسی می‌شود. البته همان‌گونه که در بخش (۵) تشریح خواهد شد، در مقایسه با روش آینده نگر جزئی، روش MFL آینده‌نگری (بررسی روبه‌جلوی) بیشتری انجام می‌دهد. به عبارت دیگر از نظر تعداد بررسیهای انجام شده در هر مرحله، روش MFL مابین دو روش PL و FL قرار دارد؛ درحالی‌که از نظر تعداد مراحل حل حداقل به خوبی FL عمل می‌کند.

در الگوریتم MFL، در شروع بررسی دامنه یک متغیر آتی مثل U_i ، مقادیری از دامنه U_i و متغیرهای آتی پس از آن که با کلیه مقادیر باقیمانده در دامنه متغیر U_{i-1} ، ناسازگارند، شناسایی شده و بدون انجام بررسیهای سازگاری اضافی، از دامنه آنها حذف می‌شوند. (همان‌گونه که در بخش بعد خواهیم دید، این عمل با به خاطر سپاری بررسیهای سازگاری انجام شده در مراحل قبلی میسر می‌شود). به علاوه، بررسی سازگاری مقدار L_i موجود در دامنه U_i با مقادیر موجود در دامنه هر یک از متغیرهای آتی بعد از آن، تا جایی صورت می‌گیرد که مقدار سازگاری با L_i پیدا شود. در صورتی‌که چنین مقداری پیدا نشود، مقدار L_i نیز از دامنه U_i حذف می‌شود. اما در صورتی‌که در دامنه هر یک از متغیرهای آتی بعد از U_i ، حداقل یک مقدار وجود داشته باشد که با مقدار L_i سازگار باشد، مقدار L_i در دامنه U_i باقی ماند. الگوریتم PL در این چنین شرایطی بررسی متغیر $Li+1$ از دامنه U_i را شروع می‌کند. لیکن در الگوریتم MFL بررسی سازگاری L_i با مقادیری از دامنه متغیرهای آتی بعد از U_i که سازگاری آنها قبلاً با مقدار L_i بررسی نشده است، ادامه می‌یابد. (در واقع این آینده‌نگری بیشتری است که الگوریتم MFL نسبت به الگوریتم PL انجام می‌دهد). در صورتی‌که L_i آخرین مقدار حذف نشده موجود در دامنه مقادیر متغیر آتی U_i باشد، از میان مقادیر فوق، آنهایی که با L_i نیز ناسازگار باشند، مقادیری‌اند که با تمام مقادیر باقیمانده در دامنه مقادیر متغیر آتی U_i ناسازگارند. در اصل نقطه‌ی قوت الگوریتم آینده‌نگر کامل بهبود یافته، تشخیص این مقادیر است؛ زیرا چنین مقادیری در طی بررسی مقادیر موجود در دامنه مقادیر متغیر آتی U_{i+1} ، شناسایی شده و بدون انجام بررسیهای

سازگاری اضافی که الگوریتم FL انجام می‌دهد، حذف می‌شوند. به عنوان مثال، شکل (۴) مراحل اجرای الگوریتم MFL را در حل مسئله چهار وزیر نشان می‌دهد. (الگوریتم MFL در بخش پیوست به صورت شبه کد آورده شده است). در مرحله اول از شکل (۴)، پس از انتخاب مقدار ۱ برای وزیر اول (متغیر Q_1)، ابتدا مقادیری از دامنه متغیرهای آتی که با این مقدار ناسازگارند، از دامنه وزیرهای بعدی حذف می‌شود (یعنی خانه‌هایی که با علامت FC مشخص شده‌اند). سپس مقدار ۳ از دامنه وزیر دوم (Q_2) نیز به علت ناسازگاری با مقادیر باقیمانده در دامنه وزیر سوم حذف می‌شود (این امر با علامت PL در خانه مربوطه نشان داده شده است). پس از اتمام بررسیهای سازگاری دامنه وزیر دوم با دامنه وزیرهای بعدی، تنها مقدار باقیمانده در دامنه وزیر دوم، مقدار ۴ است. با توجه به اینکه مقدار ۴ از دامنه وزیر سوم و مقدار ۲ از دامنه وزیر چهارم، با تنها مقدار باقیمانده برای وزیر دوم ناسازگارند، الگوریتم MFL در همین مرحله این مقادیر را نیز از دامنه وزیرهای سوم و چهارم حذف می‌کند. این امر با علامت MFL در خانه‌های مربوطه نشان داده شده است.

اگر به جای الگوریتم MFL از الگوریتم PL استفاده شود، خانه‌هایی که با علامت MFL نشان داده شده‌اند، اصلاً حذف نمی‌شوند و به همین خاطر یک مرحله‌ی اضافی (مرحله ۲ از شکل ۲) نسبت به MFL لازم است تا الگوریتم PL متوجه نامناسب بودن مقادیر انتخابی در مراحل قبلی شده و عمل عقبگرد را شروع کند. توجه کنید که هرچه تعداد وزیرهای مسئله زیادت‌تر شود، تعداد مراحل اضافی که PL نسبت MFL انجام می‌دهد، به مراتب بیشتر از مثال فوق خواهد شد. از طرف دیگر، اگر از الگوریتم FL استفاده شود، مقادیر مزبور بالاخره حذف می‌شوند شکل (۳)، ولی این حذف زمانی صورت می‌گیرد که سازگاری دامنه وزیرهای سوم و چهارم نیز با دامنه وزیر دوم بررسی شود، در حالی‌که این بررسی اضافی در MFL اصلاً انجام نمی‌شود. به عبارت دیگر، MFL در زمان بررسی دامنه وزیر دوم با وزیرهای سوم و چهارم متوجه نامناسب بودن

	۱	۲	۳	۴		۱	۲	۳	۴		۱	۲	۳	۴		۱	۲	۳	۴		۱	۲	۳	۴	
Q1	✓					BT	✓				BT	✓				BT	✓				BT	✓			
Q2	FC	FC	PL			FC	FC	FC			FC	FC	FC	✓		FC	FC	FC	✓		FC	FC	FC	✓	
Q3	FC	PL	FC	FL			FC	FL	FC			FC	FL	FC		✓	FC	FL	FC		✓	FC	FL	FC	
Q4	FC	FL		FC		FL	FC		FL		FL	FC		FL		FL	FC		FL		FL	FC	✓	FL	

شکل ۴- مراحل اجرای الگوریتم آینده‌نگر کامل بهبود یافته (MFL) در حل مسئله چهار وزیر

متغیرهای آتی نیز با یکدیگر بررسی می‌شود. لیکن از آنجایی که حاصل هیچ‌یک از بررسیهای انجام شده در یک مرحله به خاطر سپرده نمی‌شود، برخی از آنها در مراحل بعدی تکرار می‌شوند. به عنوان مثال، فرض کنید که در یک مرحله از یکی از الگوریتمهای آینده‌نگر، U متغیر جاری باشد و مقدار L_i از دامنه متغیر آتی U_i ، پس از انجام بررسی سازگاری با مقدار متغیر U_i حذف نشود. تا رسیدن به مرحله‌ای که نوبت به تعیین مقدار U_i فرا نرسیده است، در هر مرحله، سازگاری مقدار L_i (در صورت عدم حذف) با مقادیر موجود در دامنه مقادیر متغیرهای آتی دیگر مجدداً بررسی خواهد شد. در نهایت، زمانی که نوبت به تعیین مقدار U_i می‌رسد، سازگاری مقدار L_i (در صورت عدم حذف) باز هم با مقادیر موجود در دامنه مقادیر متغیرهای آتی بعد از U_i بررسی خواهد شد. برتری الگوریتم MFL نسبت به FL در این است که با به خاطر سپاری بررسیهای سازگاری انجام شده قبلی، از تکرار مجدد آنها در ادامه جستجو پرهیز می‌کند.

اگرچه هر دو الگوریتم FL و MFL در کل تعداد بررسیهای سازگاری بیشتری نسبت به الگوریتمهای FC و PL انجام می‌دهند، و تعدادی از این بررسیهای سازگاری تکراری است، لیکن الگوریتم FL، در طی هر مرحله نیز بسیاری از بررسیهای سازگاری تکراری انجام می‌دهد. به عنوان مثال، سازگاری مقدار L_1 از دامنه متغیر آتی U_1 و L_2 از دامنه U_2 ، یک بار در حین بررسی دامنه U_1 و (در صورت عدم حذف مقدار L_1) بار دیگر در حین بررسی دامنه U_2 بررسی می‌شود. اما در الگوریتم MFL، با به خاطر سپاری بررسیهای سازگاری انجام شده در هنگام بررسی دامنه متغیر آتی U_1 با دامنه متغیرهای آتی پس از

مقادیر مزبور شده و نیازی به بررسی مجدد دامنه وزیرهای سوم و چهارم با وزیر دوم (یعنی کاری که روش FL به طور عادی انجام می‌دهد) ندارد. جزئیات بیشتر در خصوص الگوریتم آینده‌نگر کامل بهبود یافته (MFL) در مرجع [۲] ارائه شده است.

۵- مقایسه الگوریتمهای جستجوی آینده‌نگر

همان‌گونه که در بخش قبل توضیح داده شد، کلیه الگوریتمهای جستجوی آینده‌نگر با حذف مقادیری از دامنه متغیرهای آتی که امکان انتخاب آنها در آینده وجود ندارد، انتخاب مقدار برای متغیر جاری را به صورت یک محدودیت انتشار می‌دهند. با این حال، میزان حذف و تلاشی که در قالب انجام بررسیهای سازگاری صورت می‌گیرد، در الگوریتمهای مختلف آینده‌نگر متفاوت است. هر چه میزان حذف مقادیر نامناسب از دامنه متغیرهای آتی بیشتر باشد، انتخابهای کمتری در هنگام تعیین مقدار متغیرهای آتی باقی خواهد ماند و در نتیجه، با تعداد عقبگرد و مراحل کمتری به حل مسئله (البته در صورتی که مسئله حل داشته باشد)، دست می‌یابیم. اما حذف مقادیر بیشتر از دامنه متغیرهای آتی نیاز به انجام بررسیهای سازگاری بیشتری دارد.

همان‌گونه که در بخش (۱-۳) توضیح داده شد، در الگوریتم FC، بررسیهای سازگاری تنها بین مقدار انتخاب شده برای متغیر جاری و مقادیر موجود در دامنه مقادیر متغیرهای آتی صورت می‌گیرد و دامنه برخی متغیرهای آتی به علت وجود ناسازگاری با مقدار متغیر جاری هرس می‌شود. در الگوریتمهای PL و FL سازگاری مقادیر موجود در دامنه مقادیر

آن، از تکرار این بررسیها در ادامه جستجو پرهیز می‌شود. بدین صورت که قبل از شروع به بررسی مقادیر موجود در دامنه متغیر UI، از قبل مشخص شده است که کدام یک از مقادیر موجود در دامنه آن و متغیرهای آتی بعد از آن، با تمامی مقادیر باقیمانده در دامنه متغیر آتی قبل از آن، ناسازگار است و بایستی حذف شوند. سایر مقادیر موجود در دامنه UI نیز همانند الگوریتم PL، تنها با مقادیر موجود در دامنه متغیرهای آتی بعد از UI بررسی می‌شوند.

بنابراین جایگاه الگوریتم MFL از نظر میزان بررسیهای سازگاری یا آینده‌نگری که انجام می‌دهد مابین PL و FL است. در واقع میزان بررسیهای سازگاری که MFL انجام می‌دهد، تنها اندکی بیشتر از PL و به مراتب کمتر از FL است. این نتیجه به صورت تجربی نیز با استفاده از یک سامانه کمک آموزشی که به منظور تسهیل آموزش روشهای حل مسائل ارضای محدودیت در دانشکده مهندسی کامپیوتر دانشگاه صنعتی شریف پیاده‌سازی شده است، بر روی چندین مسئله ارضای محدودیت با اندازه‌های مختلف تأیید شده است [۲]. نمودار شکل (۵) به عنوان یک نمونه، نتیجه مقایسه این الگوریتمها را از نظر تعداد بررسیهای سازگاری انجام شده در طی یافتن کلیه پاسخ‌های دو مسئله هشت و شانزده وزیر نشان می‌دهد (تعداد کل پاسخ‌های این دو مسئله به ترتیب برابر با ۹۲ و ۱۴۷۷۵۱۲ است). همان‌گونه که در شکل مشاهده می‌شود، الگوریتم MFL از نظر تعداد بررسیهای سازگاری که انجام می‌دهد (و در نتیجه از نظر کارایی) در بین دو الگوریتم آینده‌نگر PL و FL، به الگوریتم PL نزدیکتر است.

زمان لازم برای انجام یک بررسی سازگاری تنها در یک مسئله ارضای محدودیت مشخص، مقدار ثابتی است که بستگی به سرعت کامپیوتر استفاده شده دارد و به هر حال در همه الگوریتم‌های فوق یکسان است. لذا کل زمان لازم برای یافتن کلیه پاسخ‌های یک مسئله رابطه مستقیمی با کل تعداد بررسیهای سازگاری انجام شده در حل آن مسئله دارد. در نتیجه نمودارهای ارائه شده در شکل ۵ در واقع به طور ضمنی مقایسه تجربی الگوریتمهای آینده‌نگر با توجه به ماهیت غیرمطلع^{۲۴} و سیستماتیک^{۲۵} بودن آنها برای حل این نوع از مسائل به صورت نمایی است [۱۳]. یعنی با بزرگ شدن اندازه مسئله، زمان مورد نیاز برای حل مسئله به صورت نمایی افزایش می‌یابد. الگوریتم MFL نیز از این قاعده مستثنی نیست و در حل مسائل کلی ارضای محدودیت دارای پیچیدگی نمایی است. نمودار شکل ۵ کاملاً مبین این نکته است. همان‌گونه که در این نمودار نشان داده شده است، در حالی که اندازه‌ی مسئله شانزده وزیر تنها دو برابر مسئله هشت وزیر است، تعداد بررسیهای سازگاری لازم برای یافتن کلیه پاسخ‌های این مسئله بیش از ۶۰۰۰۰۰ برابر تعداد بررسیهای سازگاری لازم برای حل مسئله هشت وزیر است.

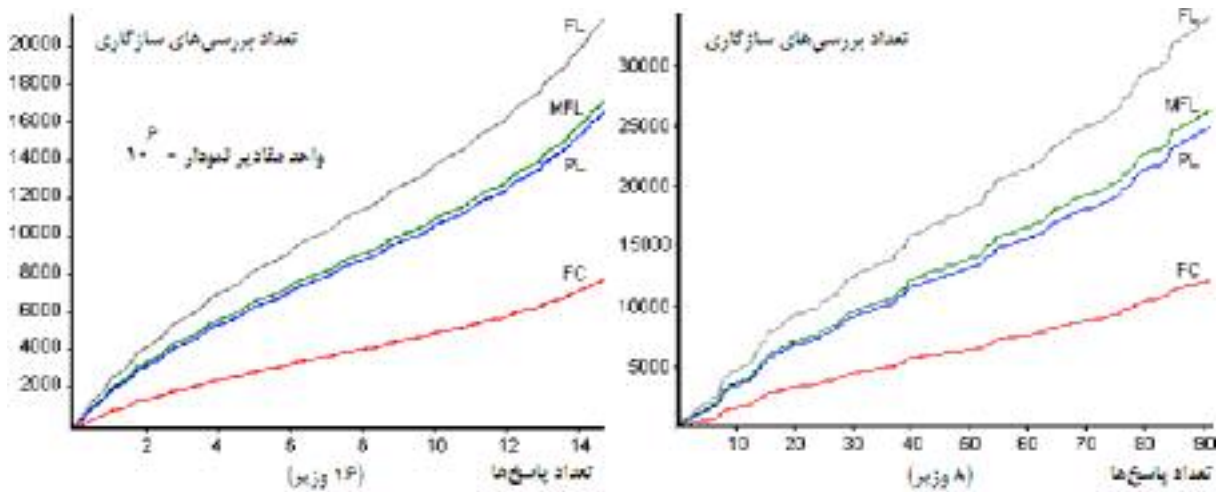
مقایسه تجربی الگوریتمهای آینده‌نگر موجود با الگوریتم MFL نشان می‌دهد که از نظر تعداد قدمهای لازم برای حل مسئله، نیز عملکرد این الگوریتم حداقل به خوبی FL و به مراتب بهتر از دو الگوریتم دیگر است. به عنوان نمونه، نمودار شکل (۶) این نکته را برای یافتن کلیه پاسخ‌های دو مسئله هشت و شانزده وزیر نشان می‌دهد. بنابراین، روش MFL در مقایسه با FL، علاوه بر انجام بررسی سازگاری به مراتب کمتر شکل (۵)، در طی مراحل کمتری نیز به راه حل مسئله (در صورت وجود) دست می‌یابد شکل (۶). البته توجه شود که برتری اصلی الگوریتم MFL در کاهش تعداد بررسیهای سازگاری اضافی است که الگوریتم FL انجام می‌دهد، و نه کاهش تعداد مراحل حل مسئله؛ هرچند که تعداد مراحل لازم برای رسیدن به حل نیز با استفاده از MFL کاهش می‌یابد.

با توجه به اینکه در دو الگوریتم FC و PL، مقادیر کمتری از دامنه مقادیر متغیرهای آتی حذف می‌شود، واضح است که به

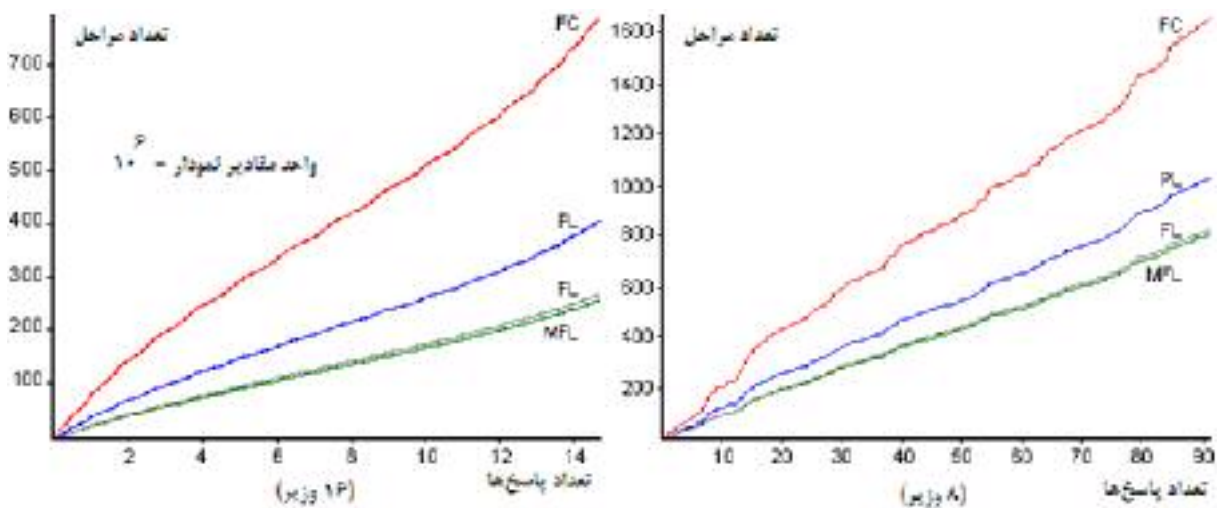
زمان لازم برای انجام یک بررسی سازگاری تنها در یک مسئله ارضای محدودیت مشخص، مقدار ثابتی است که بستگی به سرعت کامپیوتر استفاده شده دارد و به هر حال در همه الگوریتم‌های فوق یکسان است. لذا کل زمان لازم برای یافتن کلیه پاسخ‌های یک مسئله رابطه مستقیمی با کل تعداد بررسیهای سازگاری انجام شده در حل آن مسئله دارد. در نتیجه نمودارهای ارائه شده در شکل ۵ در واقع به طور ضمنی

مقایسه تجربی الگوریتمهای آینده‌نگر موجود با الگوریتم MFL نشان می‌دهد که از نظر تعداد قدمهای لازم برای حل مسئله، نیز عملکرد این الگوریتم حداقل به خوبی FL و به مراتب بهتر از دو الگوریتم دیگر است. به عنوان نمونه، نمودار شکل (۶) این نکته را برای یافتن کلیه پاسخ‌های دو مسئله هشت و شانزده وزیر نشان می‌دهد (تعداد کل پاسخ‌های این دو مسئله به ترتیب برابر با ۹۲ و ۱۴۷۷۵۱۲ است). همان‌گونه که در شکل مشاهده می‌شود، الگوریتم MFL از نظر تعداد بررسیهای سازگاری که انجام می‌دهد (و در نتیجه از نظر کارایی) در بین دو الگوریتم آینده‌نگر PL و FL، به الگوریتم PL نزدیکتر است.

زمان لازم برای انجام یک بررسی سازگاری تنها در یک مسئله ارضای محدودیت مشخص، مقدار ثابتی است که بستگی به سرعت کامپیوتر استفاده شده دارد و به هر حال در همه الگوریتم‌های فوق یکسان است. لذا کل زمان لازم برای یافتن کلیه پاسخ‌های یک مسئله رابطه مستقیمی با کل تعداد بررسیهای سازگاری انجام شده در حل آن مسئله دارد. در نتیجه نمودارهای ارائه شده در شکل ۵ در واقع به طور ضمنی



شکل ۵ - تعداد بررسی‌های سازگاری انجام شده برای یافتن کلیه پاسخهای مسائل هشت و شانزده وزیر



شکل ۶ - تعداد مراحل طی شده برای یافتن کلیه پاسخهای مسائل هشت و شانزده وزیر

بررسی‌های انجام شده قبلی، مشخص است که کدام یک از مقادیر موجود در دامنه $U1$ و متغیرهای آتی بعد از $U1$ ، با تمامی مقادیر باقیمانده در دامنه متغیر آتی ماقبل آن، ناسازگارند و بایستی حذف شوند. حذف این مقادیر باعث می‌شود که در طی بررسی سازگاری دامنه $U1$ و دامنه متغیرهای آتی بعد از آن، نه تنها بررسی‌های سازگاری کمتری با مقادیر موجود در دامنه متغیرهای آتی بعد از آنها صورت گیرد، بلکه احتمال حذف

تعداد قدمهای بیشتری برای رسیدن به کلیه پاسخهای مسئله مزبور نیاز خواهند داشت. اما همان‌گونه که نمودار شکل (۶) نشان می‌دهد، تعداد مراحل الگوریتم FL از الگوریتم MFL اندکی بیشتر است. این امر بدین معنی است که در الگوریتم MFL ، مقادیر بیشتری از دامنه مقادیر متغیرهای آتی حذف می‌شوند؛ زیرا در این الگوریتم، قبل از شروع بررسی مقادیر موجود در دامنه یک متغیر آتی مانند $U1$ ، با به خاطر سپاری

عنوان یک محدودیت جدید و انتشار آن، دامنه متغیرهای آتی را هرس می‌کنند. میزان حذف و تلاشی که در قالب انجام بررسیهای سازگاری صورت می‌گیرد، در الگوریتمهای مختلف آینده‌نگر متفاوت است. هر چقدر میزان حذف مقادیر از دامنه متغیرهای آتی بیشتر باشد، تعداد انتخاب‌های ممکن برای مقادیر آتی به متغیرهای آتی در ادامه‌ی جستجو کاهش یافته و در نتیجه، تعداد مراحل طی شده و تعداد عقبگردهای انجام شده در راه رسیدن به پاسخ مسئله کاهش می‌یابد. اما حذف مقادیر بیشتر از دامنه متغیرهای آتی، نیاز به انجام بررسیهای سازگاری بیشتری دارد. در بین الگوریتمهای آینده‌نگر موجود، روش آینده‌نگر کامل به نسبت از دو الگوریتم آینده‌نگر جزئی و بررسی جلورو تعداد بررسیهای سازگاری بیشتری انجام داده و مسائل را در طی مراحل و انجام عقبگردهای کمتری حل می‌کنند.

در این مقاله الگوریتم جستجوی آینده‌نگر جدیدی با نام "آینده‌نگر کامل بهبود یافته" معرفی شد که از الگوریتم اصلی "آینده‌نگر کامل" کاراتر است؛ بدین معنی که بررسیهای سازگاری کمتری انجام داده و در عین حال، مقادیر ناسازگار بیشتری را از دامنه متغیرهای آتی حذف می‌کند. در این الگوریتم تعداد مراحل طی شده و همچنین تعداد عقبگردهای انجام شده تا رسیدن به پاسخ مسئله نیز کمتر از الگوریتم آینده‌نگر کامل است.

مقادیر موجود در دامنه $U1$ و متغیرهای آتی بعد از آن نیز افزایش یابد. به عنوان مثال، اگر در دامنه یکی از متغیرهای آتی بعد از $U1$ مثلاً $U2$ ، دو مقدار $L21$ و $L22$ وجود داشته باشند و پس از پایان بررسیهای سازگاری مقادیر موجود در دامنه $U1$ ، مقدار $L21$ به دلیل عدم سازگاری با مقادیر باقیمانده در دامنه $U1$ حذف گردد، سازگاری مقادیر موجود در دامنه متغیرهای آتی بین $U1$ و $U2$ ، تنها با مقدار $L22$ بررسی می‌شود. اما در الگوریتم FL، پس از پایان بررسیهای سازگاری دامنه $U1$ ، مقدار $L21$ از دامنه مقادیر متغیر آتی $U2$ حذف نمی‌شود و در نتیجه، مقادیری از دامنه متغیرهای آتی بین $U1$ و $U2$ که با مقدار $L21$ سازگار ولی با مقدار $L22$ ناسازگار باشند، نیز حذف نخواهند شد. برای توضیحات بیشتر و جزئیات کلیه الگوریتمهای مورد بحث این بخش به مرجع [۲] مراجعه شود.

۶- نتیجه‌گیری

مسائل ارضای محدودیت، بخش نسبتاً وسیعی از مسائل مطرح در زمینه هوش مصنوعی را در بر می‌گیرند. این مسائل با استفاده از مجموعه‌ای از متغیرها و تعدادی محدودیت بر روی مقادیری که این متغیرها می‌توانند اختیار کنند، تعریف می‌شوند. پاسخ این مسائل مجموعه‌ای از مقادیر منحصر به فرد برای متغیرهاست، به طوری که تمامی محدودیتهای مورد نظر مسئله ارضا شده باشد.

الگوریتمهای جستجوی آینده‌نگری که ویژه حل این نوع از مسائل ابداع شده است، در ابتدای هر مرحله، مقداری را برای متغیر جاری انتخاب می‌کنند و با در نظر گرفتن این انتخاب به

واژه‌نامه

- | | | |
|-------------------------------------|----------------------------|-----------------------------|
| 1. Constraint Satisfaction Problems | 10.Unary Constraints | 19.Partially Lookahead |
| 2. Consistent Labeling Problems | 11.n-ary constraints | 20.Fully Lookahead |
| 3. Backtracking | 12.search space | 21.Modified Fully Lookahead |
| 4. Consistency Checking | 13.Initial State | 22.Binary Constraints |
| 5. Crypt Arithmetic | 14.Goal State | 23.Prune |
| 6. Eight Queen | 15.Depth First Search | 24.Uninformed |
| 7. Map Coloring | 16.Breadth First Search | 25.Systematic |
| 8. Waltz | 17.Iterative Deepening DFS | |
| 9. Consistent | 18.Forward Checking | |

۱. قاسم ثانی، غ. و نمازی، م.، "روشی برای حل مسائل برچسب دهی سازگار"، مجموعه مقالات پنجمین سمینار سالانه انجمن کامپیوتر ایران، دانشگاه شهید بهشتی، ص. ۲۲۶-۲۱۶، ۱۳۷۸.
۲. نمازی، م.، "طراحی و پیاده‌سازی کمک آموزشگر الگوریتمهای جستجوی ویژه حل مسائل ارضای محدودیت"، پایان‌نامه کارشناسی، دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف، ۱۳۷۸.
3. Bacchus, F., and Grove, A., "On the Forward Checking Algorithm," *In Proceedings of the First International Conference on Principles and Practice of Constraint Programming*, PP. 292-309, 1995.
4. Dechter, R., and Pearl, J., "Generalized Best-First Search Strategies and the Optimality of A*," *Journal of the Association for Computing Machinery*, Vol. 32, PP. 505-536, 1985.
5. Dechter, R., and Pearl, J., "Network-Based Heuristics for Constraint Satisfaction Problems," *Artificial Intelligence Journal*, Vol. 34, PP. 1-38, 1988.
6. Freuder, E., "A Sufficient Condition for Backtrack-Free Search," *Journal of the Association for Computing Machinery*, Vol. 29, PP. 24-32, 1982.
7. Haralick, R., and Elliot, G., "Increasing Tree Search Efficiency for Constraint-Satisfaction Problems," *Artificial Intelligence Journal*, Vol. 14, PP. 263-313, 1980.
8. Kanal, L., and Kumar, V., *Search in Artificial Intelligence*, Springer Verlag, 1988.
9. Korf, R., "Depth First Iterative-Deepening: an Optimal Admissible Tree Search," *Artificial Intelligence Journal*, Vol. 27, PP. 97-109, 1985.
10. Korf, R., "Optimal Path Finding Algorithms," in L. Kanal and V. Kumar ed., *Search in Artificial Intelligence*, chapter 7, PP. 223-263, Springer-Verlag, 1988.
11. Korf, R., "Linear-Space Best-First Search," *Artificial Intelligence Journal*, Vol. 62, PP. 41-78, 1993.
12. Kumar, V., "Algorithms for Constraint Satisfaction Problems: A Survey," *Artificial Intelligence Magazine*, Vol. 13, No. 1, PP. 32-44, 1992.
13. Mackworth, A., "Consistency in Network of Relations," *Artificial Intelligence Journal*, Vol. 8, PP. 99-118, 1977.
14. Nilsson, N., *Artificial Intelligence: A new Synthesis*, Morgan Kaufmann, 1998.
15. Rich, E., and Knight, K., *Artificial Intelligence*, 2nd Ed., McGraw Hill, 1991.
16. Russel, S., and Norvig, P., *Artificial Intelligence: A Modern Approach*, Prentice Hall, 1995.
17. Tsang, E., "Notes on Consistency in Consistent Labeling Problems," Technical Report CSCM-29, University of Essex, Department of Computer Science, U.K., 1987.
18. Waltz, D., *Understanding line Drawings of Scenes with Shadow*, in P. Winston ed., *The Psychology of Computer Vision*, McGraw Hill, 1975.

در این بخش الگوریتمهای بررسی جلورو (FC)، آینده‌نگر کامل بهبودیافته، و رویه‌ای به نام Forwardsearch که به عنوان برنامه اصلی (شکل پ ۱)، زیر رویه‌های مربوط به دو الگوریتم مزبور (شکلهای پ ۲ و ۳) را فراخوانی می‌کند، به صورت شبه کد ارائه می‌شود. یادآوری می‌شود که کلیه الگوریتمهای آینده‌نگر PL، FL و MFL در ابتدای هر مرحله الگوریتم بررسی جلورو (FC) را اجرا می‌کنند. لذا در اینجا نیز برای تکمیل الگوریتم MFL، الگوریتم FC نیز آورده شده است. برای اطلاعات بیشتر در خصوص این الگوریتمها به [۱ و ۷] رجوع شود.

```
Recursive Procedure Forwardsearch (U, F, D);
  For F(U)=each element of D(U) Begin
    if U < Number_Of_Variables then Begin
      New_D = Forward_Check (U, F(U), D);
      if Not Empty_Domain_Flag then Call Modified_FL (U, New_D);
      end if;
      if Not Empty_Domain_Flag then Begin
        Call Forwardsearch(U+1, F, New_D);
      end if;
    else
      Output the Labeling F;
    end for;
  end Forwardsearch ;
```

شکل پ ۱ - رویه جستجوی جلورو (رویه اصلی)

```
Function Forward_Check(U, L, D) : New Domain Table ;
  New_D : Empty Domain Table ;
  For U2=U+1 to Number_Of_Variables Begin
    For L2=each element of D(U2)
      if relation(U, L, U2, L2) then Begin
        Enter L2 into New_D(U2);
        Dellevel(U2, L2) = U+1;
      end if
    if New_D(U2) is Empty then Begin
      Empty_Domain_Flag = True ;
      return(New_D);
    end if;
  end for U2
  return(New_D);
end Forward_Check ;
```

شکل پ ۲ - زیر رویه مربوط به الگوریتم بررسی جلورو (FC)

```

Procedure Modified_FL(U, New_D);
  For U1=U+1 to Number_Of_Variables Begin
    For L1=each element of New_D(U1) Begin
      if Dellevel(U1, L1) = U1-1 then Begin
        Delete L1 from List New_D(U1)
      else Begin
        For U2=U1+1 to Number_Of_Variables Begin
          Consistent_Label_Found_Flag = False;
          For L2=each element of New_D(U2) Begin
            if Dellevel(U2, L2) = U1-1 then
              Delete L2 from List New_D(U2) ;
            else if relation(U1, L1, U2, L2) then begin
              Consistent_Label_Found_Flag = True;
              Lastchecked(U2) = L2 ;
              Break For L2 Loop;
            end if;
          end For L2 Loop;
          if Not Consistent_Label_Found_Flag then begin
            Delet L1 from list New_D(U1);
            Break For U2 Loop;
          end if ;
        end For U2 Loop;
        if Consistent_Label_Found_Flag then begin
          For U2=U1+1 to Number_Of_Variables Begin
            Dellevel(U2,LastChecked(U2)) = U1+1 ;
            For L2 = each remained element of D(U2) that not checked Begin
              if Dellevel(U2,L2) = U1-1 then
                Delete L1 from List New_D(U2) ;
              else if Dellevel(U2, L2)=U1 then
                if relation(U1, L1, U2, L2) then
                  Dellevel(U2, L2) = U1+1 ;
                end For L2 Loop
            end For U2 Loop;
          end if ;
        end of else ;
      end For L1 Loop;
      if New_D(U1) is empty then begin
        Empty_Domain_Flag=True ;
        return;
      end if ;
    end For U1 Loop ;
  return;
end Modified_FL;

```

شکل پ ۳ - زیر رویه مربوط به الگوریتم آینده نگر کامل بهبود یافته (MFL)