دانشکدگان علوم
دانشکده ریاضی، آمار و علوم کامپیوتر

بهبود روش وارسی مدل با استفاده از نظریه تعبیر مجرد

نگارنده

پویا پرتو

استاد راهنمای اول: دکتر مجید علی‌زاده
استاد راهنمای دوم: دکتر مجتبی مجتهدی

پایاننامه برای دریافت درجه کارشناسی ارشد
در رشته علوم کامپیوتر

تاریخ دفاع

چکیده

روش وارسی مدل یک روش قابل اعتماد برای بررسی صحت عملکرد برنامه‌های کامپیوتری است. بیان‌های مختلف این روش از منطق موجهات بهره می‌برند که چندان برای برنامه نویسان شناخته شده نیستند. در این رساله سعی شده یک بیان جدید از روش وارسی مدل مورد شرح و بررسی قرار گیرد که در آن به کمک نظریه تعبیر مجرد به جای منطق موجهات از عبارات منظم استفاده شده است.

پس از ارائه‌ی مفاهیم اولیه، به سه صورت متفاوت به بیانی جدید از روش وارسی مدل پرداخته‌ایم. صورت اول ساختار خاصی ندارد و صرفا در ادبیات نو بیان شده است، صورت دوم ساختار عبارات منظم را به صورتبندی‌اش اضافه کرده است و در صورت سوم با اضافه شدن ساختار برنامه به صورتبندی روشْ به پیاده سازی نزدیک‌تر شده است.
معادل بودن این سه صورت نیز مطالعه و بررسی می‌شود.

کلمات کلیدی: وارسی مدل، نظریه تعبیر مجرد، معناشناسی دلالتی، پیوند گالوا، درستی یابی صوری، تحلیل ایستا، درستی یابی برنامه‌های کامپیوتری

تقدیم به

تقدیم به

سپاسگزاری

سپاسگزاری

پیشگفتار

با توجه به پیشرفت روز افزون علوم کامپیوتر و ورود کاربردهای آن به زندگی روزمره، پیشرفت در روش‌های ساخت و نگه‌داری برنامه‌ها نیازی آشکار به نظر می‌رسد. یکی از مسائل مهم در این زمینه بررسی صحت کارکرد برنامه‌های نوشته‌شده است. عدم صحت کارکرد برنامه‌های نوشته شده بسته به حساسیت یک برنامه می‌تواند تبعات زیان‌بار جبران ناپذیری به همراه داشته‌باشد. پرتاب ناموفق آریان ۵[۱۸] ، از مدار خارج شدن مدارگرد مریخ [۲] و تصادف هلیکوپتر چینوک [۱] چند نمونه از تبعات بزرگ این قضیه در گذشته بوده اند، همین‌طور به‌سادگی می‌توان فجایع دیگری از این دست را در زندگی روزمره‌ی انسان‌ها متصور شد. برای تعیین صحت کارکرد برنامه‌های کامپیوتری روش های متفاوتی ابداع شده‌اند که در ادامه به طور مختصر از آن‌ها یاد می‌کنیم اما پیش از آن به یک خاصیت مشترک همه‌ی این روش‌ها، یعنی "ناکامل بودن"، می‌پردازیم. منظور از ناکامل بودن این است که با استفاده از هیچ یک از روش‌هایی که داریم نمی‌توانیم هر خاصیتی را برای هر برنامه‌ای بررسی کنیم. به عبارت دیگر استفاده از هر روشی، محدودیت‌هایی دارد. و البته قضیه رایس [۲۲] به ما این تضمین را داده که روش کاملی اصلا وجود ندارد. قضیه رایس(به طور غیر رسمی) بیان می‌کند که مسئله ی بررسی هر خاصیت غیر بدیهی، برای همه ی برنامه‌ها، تصمیم ناپذیر است. این دلیلی بر این شده که روش‌های مختلفی برای این کار معرفی شوند که هر کدام می‌توانند حالت‌های خاصی از مسئله را حل کنند.

یک دسته‌بندی برای این روش‌ها تقسیم آن‌ها به دو دسته‌ی پویا و ایستا است. روش‌های پویا روش‌هایی هستند که در آن‌ها تست برنامه همزمان با اجرای برنامه است، درحالیکه روش‌های ایستا بدون اجرای برنامه‌ها آن‌ها را تست می‌کنند.
روش‌های پویا معمولاً با اجرای حالات محدودی از برنامه، تصمیم می‌گیرند که برنامه‌ای که نوشته شده، انتظارات را برآورده می‌کند یا خیر. اگر این روش بتواند تشخیص دهد برنامه‌ای درست کار نمی‌کند، می‌توانیم با اطمینان نتیجه بگیریم که برنامه غلط نوشته‌شده است، اما اگر برنامه‌ای از تست‌های ساخته‌شده با این روش‌ها با موفقیت عبور کند، نمی‌توان اطمینان حاصل کرد که برنامه درست کار می‌کند، زیرا ممکن است حالتی مشکل‌زا از اجرای برنامه وجود داشته باشد که در تست ها نیامده باشد. برای اطلاعات بیشتر به [۲۰] مراجعه کنید.

روش‌های ایستا معمولاً روش‌هایی هستند که از نظریه‌های مختلف در منطق ریاضی به عنوان ابزار بهره می‌برند تا بدون اجرای خود برنامه‌ها در مورد صحت اجرای آن‌ها نتیجه‌گیری کنند. به همین دلیل به بخشی مهم و بزرگی از این دستورات که از منطق استفاده می‌کنند روش‌های صوری هم گفته‌می‌شود. معروف ترین روش‌های ایستا؛ روش وارسی مدل، روش‌های استنتاجی و استفاده از نظریه تعبیر مجرد است.

در روش وارسی مدل، یک مدل صوری متناهی از برنامه‌ی موردبررسی می‌سازیم که همه‌ی حالات اجرای برنامه با آن قابل‌توصیف است، سپس با استفاده از یک زبان صوری که بتواند در مورد مدل هایمان صحبت‌کند، ویژگی‌های مورد بررسی را بیان‌می‌کنیم و در نهایت صحت ویژگی‌های بیان‌شده را بررسی می کنیم. مقاله [۴] شروع این روش‌ها بوده که این کار را با استفاده از نوعی مدل کریپکی [۱۷] و نوعی منطق زمانی به نام منطق زمانی خطی [۴] انجام داده که روشی است با دقت و البته هزینه‌ی محاسباتی بسیار بالا. [۱۲] یک منبع بسیار مقدماتی و کتاب[۵] یک مرجع سنتی در این زمینه است.

در روش‌های استنتاجی که شاید بتوان یکی از ابتدایی ترین آن‌ها را استفاده از منطق هور[۱۱] دانست، درستی کارکرد برنامه‌هایمان را با ارائه‌ی یک درخت اثبات در یک دستگاه استنتاجی، متناسب با زبان برنامه‌هایمان، نشان می‌دهیم. در این روش هم اگر بتوانیم درستی یک برنامه را اثبات کنیم، دیگر به‌طور تئوری، خیالی آسوده از درستی برنامه خواهیم داشت، اما ساختن درخت اثبات در یک نظریه برهان می‌تواند چالش برانگیز باشد چرا که این یک مسئله‌ی NP-Hard است. در[۱۲] به منطق هور به طور مقدماتی پرداخته شده است. همین‌طور کتاب[۲۱] نیز به پیاده‌سازی منطق هور در زبان coq پرداخته است، که در آن coq یک اثبات‌یار است که بر اساس نظریه نوع وابسته کار می‌کند. برای اطلاعات بیشتر در مورد چگونگی طرز کار این اثبات‌یار و نظریه‌ی بنیادین آن به کتاب[۳] مراجعه کنید. نظریه‌ی مورد شرح در[۱۰] نیز می‌تواند در این مسیر به کار گرفته شود.

نظریه تعبیر مجرد[۸] نیز یک نظریه ریاضیاتی است که به‌نوعی سعی می‌کند از روی معناشناسی یک برنامه‌ی کامپیوتری[۲۴] یک تقریب بسازد. منظور از تقریب یک دستگاه کوچک‌تر از معناشناسی اصلی است که رفتارش زیرمجموعه‌ی رفتارهای دستگاه اصلی است. سعی بر این است که دستگاه جدیدی که می‌سازیم به لحاظ محاسباتی ساده‌تر از معناشناسی اصلی کار کند تا بتوان خواص آن را راحت‌تر بررسی کرد. در این صورت هر نتیجه‌ای در مورد خواص جدید، را می‌توان برای خود برنامه هم بیان کرد، اما توجه شود که در این صورت ممکن است به همه‌ی حقایق دست پیدا نکنیم. برای اطلاعات بیشتر به [۷] و [۱۴] مراجعه شود

فهرست مطالب

فصل ۱

مقدمه

در این فصل به عنوان مقدمه‌ی کار اندکی در مورد روش وارسی مدل و نظریه‌ی تعبیر مجرد صحبت می‌کنیم و فعلا کار خاصی با [۶] که قرار است محور کار ما باشد، نداریم. بعضی از مفاهیمی که در این فصل بیان می‌شود مستقیما و برخی غیر مستقیم به بحث اصلی کار مربوط است.

۱.۱ روش وارسی مدل

روش وارسی مدل یک روش صوری است که برای درستی‌یابی سیستم‌های مختلف استفاده می‌شود. در این روش معمولا ابتدا یک ماشین حالات متناهی از روی سیستم مورد بررسی ساخته می‌شود، سپس بررسی‌هایی که قرار است روی سیستم اصلی انجام شوند، روی این ماشین(مدل) انجام می‌شود. در بررسی صحت کارکرد برنامه‌های کامپیوتری از این روش استفاده می‌شود اما این تنها مورد استفاده از این روش نیست و هر منظومه‌ی دیگری که قابلیت بیان به صورت صوری را داشته باشد قابل بررسی با این روش هست. مثلا می‌توان از این روش برای بررسی صحت عملکرد برنامه‌ی قطارهای شهری استفاده کرد؛ در حالتی که مثلا خصوصیات مورد بررسی ما عدم رخ دادن تصادف بین قطارها یا پوشش تمام مناطق شهر باشد. مثال های دیگر استفاده‌ی این روش در علوم کامپیوتر می تواند بررسی صحت عملکرد یک پردازنده یا مثلا الگوریتم تقسیمِ وظایف یک سیستم عامل باشد. این مثال‌ها هیچ یک برنامه‌ی کامپیوتری نیستند(هر چند که ممکن است مجبور باشیم از یک برنامه‌ی کامپیوتری برای پیاده سازی آن‌ها کمک بگیریم که در آن صورت بررسی صحت عملکرد آن برنامه‌ی کامپیوتری داستانی دیگر خواهد داشت) اما قابل بیان به صورت صوری به جای زبان طبیعی هستند.

ایده‌ی روش وارسی مدل از منطق‌های زمانی مختلف استفاده می‌کند. منطق زمانی یک نوع منطق موجهات است. منطق‌های موجهات از گسترش زبان منطق کلاسیک با اضافه کردن ادات وجهی گوناگون، با معانی متفاوتی که ممکن است در زبان طبیعی داشته باشند، ساخته می‌شوند.

این ادوات غالبا در زبان طبیعی نقش قید را دارند. منطق‌های زمانی بخشی از منطق‌های موجهات هستند که به صوری‌گری ما مفهوم زمان را هم اضافه می‌کنند، یعنی قیدهایی مانند فعلا، بعدا و قبلا. منطقی که در اینجا بیان می‌کنیم LTL نام دارد که یکی از منطق‌های زمانی است که برای روش وارسی مدل استفاده می‌شود. البته در مورد قیدهایی که نام بردیم ذکر این نکته ضروری است که در این بیانی که ما در اینجا از این منطق ارائه داده‌ایم ادوات جدید این فعل‌ها نیستند، هرچند که به کمک ادوات جدید می‌توان ادواتی برای هر یک از این قیود ساخت. این تکه از [۱۹] آورده شده.

ابتدا زبان این منطق را بیان می‌کنیم و سعی می‌کنیم به طور غیر دقیق در مورد معنای فرمول‌های این زبان به خواننده یک درک شهودی بدهیم؛ سپس به سراغ معناشناسی صوری این منطق می‌رویم.

۱.۱.۱ زبان **LTL**

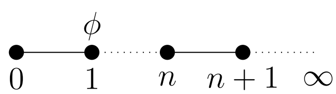
تعریف ۱.۱. هر عضو مجموعه‌ی Φ یک فرمول در زبان LTL است(و Π مجموعه‌ی(شمارای نامتناهی) فرمول‌های اتمی است و $\pi \in \Pi$):

$$\Pi \subset \Phi,$$

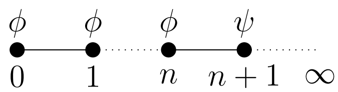
$$\phi \in \Phi \Leftrightarrow \phi ::= \pi | \phi \vee \phi | \neg\phi | \bigcirc \phi | \phi \mathcal{U} \phi$$

اولین نکته‌ای که برای فرمول‌های این زبان به آن نیاز داریم این است که در این منطق ما زمان را با اعداد طبیعی و هر خاصیتی که در موردشان تعریف شده نشان می‌دهیم. یعنی برای یک فرمول، زمان از عدد ۰ شروع شده و تا ابد ادامه خواهد داشت و حین گذر زمان ممکن است ارزش فرمول‌ها تغییر کند. مسلما پس از بررسی معناشناسی صوری بهتر می‌شود این مفهوم را به طور شهودی حس کرد، اما به هر حال به خواننده پیشنهاد می‌شود پیش از رسیدن به آن بخش به ادامه‌ی این بخش هم توجه شود.

در این زبان ادوات کلاسیک $\neg, \vee$ هستند با همان معنایی که در منطق گزاره‌ای کلاسیک داشتند. در ادوات جدید $\bigcirc\phi$ به معنای برقرار بودن این فرمول دقیقا در لحظه‌ی بعدی(دقیقا یک لحظه) است، مثلا در شکل زیر با در نظر گرفتن اینکه در زمان ۰ هستیم، این فرمول در لحظه‌ی ۱ برقرار است.



$\phi \mathcal{U} \psi$ به این معنی است که فرمول سمت چپی حداقل تا قبل از اینکه فرمول سمت راستی برقرار شود، برقرار است.(مثلا اگر بگوییم "تا وقتی که باران نباریده زمین خشک است" در این صورت "زمین خشک است" به جای فرمول سمت چپ و "باران باریده است" فرمول سمت راست است).



برای این زبان همان‌طور که در منطق کلاسیک از ادوات عطف و شرطی با اینکه با ادواتی که داریم قابل بیان هستند استفاده می‌کنند، ادات بیشتری هم هستند که مورد استفاده قرار می‌گیرند و با همین دو اداتی که معرفی کردیم قابل بیان هستند اما ما در اینجا از آن‌ها اسمی نیاورده‌ایم. دلیل وجود این‌ها هم راحت‌تر کردن کار کسی است که قرار است یک خاصیت را به صورت یک فرمول در این زبان بیان کند. همان طور که استفاده نکردن از یا و شرطی در منطق گزاره‌ای می‌تواند به سخت کردن بیان جملات در چارچوب این منطق منجر شود، حذف این ادوات وجهی هم بیان خواص را در این منطق مشکل می‌سازد. اما از آنجاییکه ما صرفا این بخش را برای معرفی این ایده قرار داده‌ایم، دیگر به بیان ادات وجهی دیگر نپرداخته‌ایم.
حال که به درکی شهودی از معنای فرمول‌های این زبان رسیده‌ایم، به بیان صوری این مفاهیم می‌پردازیم.

۲.۱.۱ معناشناسی LTL

مدل‌های این منطق را به صورت توابع $M : \mathbb{N}_0 \to P(\Pi)$ تعریف می‌کنیم. یعنی هر مدل یک تابع است که هر عدد طبیعی را به یک مجموعه از فرمول‌های اتمی می‌برد. این در واقع قرار است به این معنی باشد که یک مدل به تعبیری به این معنی است که در هر لحظه کدام یک از فرمول‌های اتمی درست هستند. مثلا در مدلی به نام M_i در واقع $M_i(5)$ مجموعه‌ی اتم‌هایی است که در لحظه‌ی ۵ طبق این مدل درست هستند و اگر اتمی در این مجموعه نباشد ارزش غلط دارد. درستی یک فرمول در یک مدل را با M, i نشان می‌دهیم و $M, i \models \phi$ به این معنی است که در لحظه‌ی iدر مدل ϕ ارزش درست دارد. این مفهوم را به صورت بازگشتی به شکل زیر تعریف می‌کنیم:

$$
\begin{array}{l}
M, i \models \pi \quad \textit{iff} \quad \pi \in M(i) \\
M, i \models \neg\phi \quad \textit{iff} \quad M, i \not\models \phi \\
M, i \models \phi \vee \psi \quad \textit{iff} \quad M, i \models \phi \quad or \quad M, i \models \psi \\
M, i \models \bigcirc\phi \quad \textit{iff} \quad M, i+1 \models \phi \\
M, i \models \phi\mathcal{U}\psi \quad \textit{iff} \quad \exists k \geq i \in \mathbb{N}_0 : \forall i \leq j < k : M, j \models \phi \quad and \quad M, k \models \psi
\end{array}
$$

برای یک فرمول اگر مدلی وجود داشته باشد که در آن مدل آن فرمول صادق باشد، آنگاه آن فرمول را ارضاپذیر می‌گوییم. اگر یک فرمول در هر مدلی صادق باشد، آن فرمول را معتبر می‌گوییم. شیوه‌ی دیگری برای بیان همین معناشناسی که گفتیم به شکل اتوماتا است.

۲.۱ نظریه تعبیر مجرد

فصل ۲

برخی مفاهیم اولیه

از اینجا کار شروع می‌شود. محوریت کار ما قرار است[۶] باشد که در آن سعی شده روش وارسی مدل با کمک نظریه تعبیر مجرد، بهبود داده شود. در[۴] روشی که معرفی شده درواقع ویژگی برنامه‌ها را به کمک منطق های LTL یا CTL بیان می‌کند. خود برنامه‌ها هم با کمک معناشناسی این منطق ها که نوع خاصی از مدل های کریپیکی به اسم سیستم گذار هستند توصیف می شوند. اما در[۶] کاری که انجام شده به این شکل است که منطق های LTL و CTL با عبارات منظم[۱۵] جایگزین شده‌اند. علت این کار بنا به ادعای نویسنده‌ی [۶]دو نکته بوده، اولی اینکه استفاده از عبارات منظم به جای منطق های نام برده شده می‌تواند برای برنامه نویسان ساده‌تر باشد و دومی اینکه عبارت منظم می‌تواند خواصی را بیان کند که منطق زمانی نمی‌تواند و [۲۵] را به عنوان شاهد ادعای دوم آورده. در ادامه ی کار وارسی مدل با استفاده از موجودات جدید تعریف شده به سه شکل مختلف بیان شده. در هر مرتبه بیان وارسی مدل، به‌گفته‌ی نویسنده، "ساختارمند"تر شده. می‌توان دریافت که فایده ساختارمندتر بودن بیان این است که پیاده‌سازی راحت‌تری دارد.

۱.۲ نحو زبان مورد بررسی

زبان بیان برنامه ها زیرمجموعه ای از دستورات زبان C است، به شکل زیر:

$$\mathsf{x}, \mathsf{y}, \ldots \in \mathbb{X}$$

$$\mathsf{A} \in \mathbb{A} ::= 1 \mid \mathsf{x} \mid \mathsf{A}_1 - \mathsf{A}_2$$

$$\mathsf{B} \in \mathbb{B} ::= \mathsf{A}_1 < \mathsf{A}_2 \mid \mathsf{B}_1 \ \mathsf{nand}\ \mathsf{B}_2$$

$$\mathsf{E} \in \mathbb{E} ::= \mathsf{A} \mid \mathsf{B}$$

$$
\begin{aligned}
\mathsf{S} \in \mathbb{S} ::= \\
&\quad \mathsf{x} \doteq \mathsf{A}; \\
&| \;\; ; \\
&| \;\; \mathsf{if}\ (\mathsf{B})\ \mathsf{S} \mid \mathsf{if}\ (\mathsf{B})\ \mathsf{S}\ \mathsf{else}\ \mathsf{S} \\
&| \;\; \mathsf{while}\ (\mathsf{B})\ \mathsf{S} \mid \mathsf{break}; \\
&| \;\; \{\mathsf{Sl}\} \\
\mathsf{Sl} \in \mathbb{SL} ::= \mathsf{Sl}\ \ \mathsf{S} \mid \epsilon \\
\mathsf{P} \in \mathbb{P}\ ::= \mathsf{Sl}
\end{aligned}
$$

در اینجا زیر مجموعه‌ای از دستورات زبان C را داریم. همین‌طور که قابل مشاهده است این زبان تا حد ممکن کوچک شده. علت این کار را بعدا عمیق‌تر حس خواهیم‌کرد. علتْ ساده‌تر شدن کار برای ارائه‌ی معناشناسی و تعبیر مجرد آن است. در اینجا راحتی آن برنامه‌نویسی که قرار است با این زبان برنامه بنویسد مطرح نبوده چون اصلا این زبان برای این کار ساخته نشده. نویسنده‌ی [۶] در اینجا صرفا می‌خواسته فرآیند را نشان دهد. اگر به فرض برای زبانی مانند پایتون بخواهیم درستی‌یابی با استفاده از روش ارائه شده را درست کینم، می‌توانیم همه‌ی راهی که در [۶] برای زبان توصیف شده، رفته‌شده را برای پایتون هم برویم و به یک تحلیل‌گر ایستا برای پایتون برسیم. در مورد قدرت بیان این زبان هم می‌توانیم بگوییم که می‌توانیم باقی اعداد را از روی عدد ۱ و عملگر منها بسازیم. مثلا ابتدا ۰ را به کمک ۱ ـ ۱ می سازیم و سپس با استفاده از ۰ می‌توانیم یکی یکی اعداد منفی را بسازیم و سپس بعد از آن به سراغ اعداد مثبت می‌رویم که با کمک ۰ و هر عدد منفی‌ای که ساختیم، ساخته می شوند. باقی اعداد و حتی باقی عملگرها(یعنی به غیر از اعداد طبیعی) نیز از روی آنچه داریم قابل‌ساختن است. در مورد عبارت‌های بولی نیز داستان به همین منوال است. یعنی اینجا صرفا ادات شفر تعریف شده و باقی عملگرهای بولی را می‌توان با استفاده از همین عملگر ساخت. باقی دستورات نیز دستورات شرط و حلقه هستند. باقی دستورات قرار است مطابق چیزی که از زبان C انتظار داریم کار بکنند. در مورد دستور ;break ذکر این نکته ضروری است که اجرای آن قرار است اجرای برنامه را از دستوری بعد از داخلی‌ترین حلقه‌ای که ;break داخلش قرار دارد ادامه دهد. در پایان می توان ثابت کرد که این زبان هم قدرت با مدل دیویس[۹] است. توجه داریم که هرچه که در این بخش درمورد معنای دستورات این زبان گفتیم، به هیچ وجه صوری نبود و صرفا درک شهودی ای که از معنای اجرای هریک از دستورات داشتیم را بیان کردیم. بیان صوری معنای برنامه‌ها را، که برخلاف درک شهودی‌مان قابل انتقال به کامپیوتر است، در ادامه بیان خواهیم‌کرد. طبیعتا این بیان صوری از روی یک درک شهودی ساخته شده‌است.

۲.۲ معناشناسی زبان مورد بررسی

معناشناسی زبانی را که در بخش پیش آوردیم با کمک مفاهیمی به نام برچسب و رد پیشوندی و عملگر چسباندن روی دو رود پیشوندی مختلف تعریف خواهیم‌کرد و نام این معناشناسی نیز معناشناسی رد پیشوندی است.

۱.۲.۲ برچسب‌ها

باوجود اینکه خود زبان C در قسمتی از زبان خود چیزهایی به نام برچسب دارد اما همین‌طور که در بخش پیشین دیدیم، در زبانی که اینجا در حال بحث روی آن هستیم خبری از برچسب‌ها نیست. اما برای تعریف صوری معنای برنامه‌ها، به شکلی که مورد بحث است، به آن‌ها نیاز است. در این بخش ابتدا به توضیحی مختصر در مورد برچسب‌ها در معناشناسی زبان مورد بحث می‌پردازیم. تعاریف صوری دقیق این موجودات در پیوست [۶] آورده‌شده‌اند. از آوردن مستقیم این تعاریف در اینجا خودداری کرده‌ایم. البته در مورد معنای صوری برچسب‌ها هم ذکر این نکته ضروری است که نویسنده‌ی [۶] حتی به صورت صوری هم برای هر بخش از برنامه این کار را به طور دقیق انجام نداده و انجام این کار به طور دقیق‌تر را احتمالا به کسی که قرار است یک پیاده سازی کامل از این روش داشته باشد سپرده.

در زبانمان Sها بخشی از موجودات موجود در زبان هستند. برچسب ها را برای Sها تعریف می‌کنیم. برچسب‌ها با کمک توابع labs, in, brks-of, brk-to, esc, aft, at تعریف می‌شوند. درواقع هر S به ازای بعضی از این توابع یک برجسب دارد و این‌ها درواقع نشان‌دهنده‌ی آن برچسب هستند. بعض دیگر این توابع برای هر S ممکن است یک مجموعه از برچسب‌ها را تعیین کند و یکی از آن‌ها هم با گرفتن S یک مقدار بولی را برمی‌گرداند.

$\text{at}[\![S]\!]$: برچسب شروع S.

$\text{aft}[\![S]\!]$: برچسب پایان S، اگر پایانی داشته باشد.

$\text{esc}[\![S]\!]$: یک مقدار بولی را بازمی‌گرداند که بسته به اینکه در S دستور ;break وجود دارد یا خیر، مقدار درست یا غلط را برمی‌گرداند.

$\text{brk-to}[\![S]\!]$: برچسبی است که اگر حین S دستور ;break اجرا شود، برنامه از آن نقطه ادامه پیدا می کند.

$\text{brks-of}[\![S]\!]$: مجموعه‌ای از برچسب ;break های S را برمی‌گرداند.

$\text{in}[\![S]\!]$: مجموعه‌ای از تمام برچسب‌های درون S را برمی‌گرداند.

$\text{labs}[\![S]\!]$: مجموعه‌ای از تمام برچسب‌هایی که با اجرای S قابل دسترسی هستند را برمی‌گرداند.

۲.۲.۲ رد پیشوندی

پس از تعریف برچسب‌ها به سراغ تعریف رد پیشوندی می‌رویم. پیش از آن باید وضعیت‌ها و محیط‌ها را تعریف کنیم.

تعریف ۱.۲. (محیط): به ازای مجموعه مقادیر $\mathbb{V}$ و مجموعه متغیرهای $\mathbb{X}$ تابع $\rho : \mathbb{X} \to \mathbb{V}$ را یک محیط می‌گوییم. مجموعه‌ی همه‌ی محیط‌ها را با $\mathbb{EV}$ نمایش می‌دهیم.

تعریف ۲.۲. (وضعیت): به هر زوج مرتبْ به ترتیب متشکل از یک برچسب l و یک محیط ρ یک وضعیت (یا حالت) $\langle l, \rho\rangle$ می‌گوییم. مجموعه‌ی همه‌ی وضعیت‌ها را با $\mathfrak{S}$ نشان می‌دهیم.

تعریف ۳.۲. (رد پیشوندی): به یک دنباله از وضعیت‌ها(با امکان تهی بودن) یک رد پیشوندی می‌گوییم.

هر رد پیشوندی یک دنباله است که قرار است توصیفی از چگونگی اجرای برنامه باشد. وضعیت‌ها همان‌طور که از نامشان پیداست قرار است موقعیت لحظه‌ای برنامه را توصیف کنند. l قرار است برچسب برنامه‌ی در حال اجرا باشد و ρ مقدار متغیرها را در آن موقع از اجرای برنامه نشان می‌دهد. دنباله‌های ما می‌توانند متناهی یا نامتناهی باشند. مجموعه‌ی ردهای پیشوندی متناهی را با $\mathfrak{S}^{+}$ و مجموعه‌ی ردهای پیشوندی نامتناهی را با $\mathfrak{S}^{\infty}$ نمایش می‌دهیم. مجموعه‌ی همه‌ی ردهای پیشوندی را هم با $\mathfrak{S}^{+\infty}$ نمایش می‌دهیم. باتوجه به آنچه گفتیم، یک عملگر چسباندن $\Join$ را روی ردهای پیشوندی تعریف می‌کنیم.

تعریف ۴.۲. (عملگر چسباندن): اگر داشته باشیم $\pi_1, \pi_2 \in \mathfrak{S}^{+\infty}, \sigma_1, \sigma_2 \in \mathfrak{S}$ داریم:

اگر $\pi_1 \in \mathfrak{S}^{+}$ داریم	$\pi_1 \Join \pi_2 = \pi_1$
اگر $\sigma_1 \neq \sigma_2$	$\pi_1 \Join \pi_2$ تعریف نشده است
اگر $\pi_1 \in \mathfrak{S}^{\infty}$ داریم	$\pi_1\sigma_1 \Join \sigma_1\pi_2 = \pi_1\sigma_1\pi_2$

همینطور ϵ هم یک رد پیشوندی است که حاوی هیچ وضعیتی نیست. به عبارت دیگر یک دنباله‌ی تهی است.

۳.۲.۲ تعریف صوری معناشناسی رد پیشوندی

در این بخش قرار است دو تابع $\mathcal{A}$ و $\mathcal{B}$ را به ترتیب روی عبارات حسابی و بولی زبانمان یعنی Aها و Bها تعریف کنیم سپس با کمک آنها S^* را روی مجموعه‌ای از اجتماع معنای Sها و Slها تعریف می کنیم. پس در نهایت هدف ما تعریف S^* است.

تعریف ۵.۲. (معنای عبارات حسابی ـ تابع $\mathcal{A}$): تابع $\mathcal{A} : \mathbb{A} \to \mathbb{EV} \to \mathbb{V}$ را به صورت بازگشتی روی ساختار $\mathsf{A} \in \mathbb{A}$ به شکل زیر تعریف می‌کنیم:

$$\mathcal{A}[\![1]\!]\rho = 1$$

$$\mathcal{A}[\![\mathsf{x}]\!]\rho = \rho(\mathsf{x})$$

$$\mathcal{A}[\![\mathsf{A}_1 - \mathsf{A}_2]\!]\rho = \mathcal{A}[\![\mathsf{A}_1]\!]\rho - \mathcal{A}[\![\mathsf{A}_2]\!]\rho$$

تعریف ۶.۲. (معنای عبارات بولی ـ تابع $\mathcal{B}$): تابع $\mathcal{B} : \mathbb{B} \to \mathbb{EV} \to \mathbb{BOOL}$ را به صورت بازگشتی روی ساختار $\mathsf{B} \in \mathbb{B}$ به شکل زیر تعریف می‌کنیم:

$\mathcal{B}[\![\mathsf{A}_1 < \mathsf{A}_2]\!]\rho = True$ اگر $\mathcal{A}[\![\mathsf{A}_1]\!]\rho$ کوچکتر از $\mathcal{A}[\![\mathsf{A}_2]\!]\rho$ باشد

$\mathcal{B}[\![\mathsf{A}_1 < \mathsf{A}_2]\!]\rho = False$ اگر $\mathcal{A}[\![\mathsf{A}_1]\!]\rho$ بزرگتر از $\mathcal{A}[\![\mathsf{A}_2]\!]\rho$ باشد

$$\mathcal{B}[\![\mathsf{B}_1 \mathsf{nand} \mathsf{B}_2]\!]\rho = \neg(\mathcal{B}[\![\mathsf{B}_1]\!]\rho \wedge \mathcal{B}[\![\mathsf{B}_2]\!]\rho)$$

طبعا $\wedge$ و $\neg$ در فرازبان هستند.

در ادامه به تعریف $\mathcal{S}^*$ می‌پردازیم. این کار را با تعریف $\mathcal{S}^*$ روی هر ساخت S و Sl انجام می‌دهیم. پیش از ادامه‌ی بحث باید این نکته را درمورد علامت‌گذاری‌هایمان ذکر کنیم که منظور از $\mathsf{S} ::= l\mathsf{break};$ این است که تاکید کرده‌ایم که S با برچسب l شروع شده‌است وگرنه همین طور که گفتیم l جزو زبان نیست.

تعریف ۷.۲. (معنای برنامه‌ها ـ تابع $\mathcal{S}^*$): اگر $\mathsf{S} ::= \mathsf{break};$ باشد، ردهای پیشوندی متناظر با اجرای این دستور را به شکل مجموعه‌ی زیر تعریف می کنیم:

$$\mathcal{S}^*[\![\mathsf{S}]\!] = \{\langle at[\![\mathsf{S}]\!], \rho\rangle | \rho \in \mathbb{EV}\} \cup \{\langle at[\![\mathsf{S}]\!], \rho\rangle\langle brk - to[\![\mathsf{S}]\!], \rho\rangle | \rho \in \mathbb{EV}\}$$

اگر $\mathsf{S} ::= \mathsf{x} \doteq \mathsf{A};$ باشد، ردهای پیشوندی متناظر با اجرای این دستور را به شکل مجموعه‌ی زیر تعریف می کنیم:

$$\mathcal{S}^*[\![\mathsf{S}]\!] = \{\langle at[\![\mathsf{S}]\!], \rho\rangle | \rho \in \mathbb{EV}\} \cup \{\langle at[\![\mathsf{S}]\!], \rho\rangle\langle aft[\![\mathsf{S}]\!], \rho[\mathsf{x} \leftarrow \mathcal{A}[\![\mathsf{A}]\!]\rho]\rangle | \rho \in \mathbb{EV}\}$$

اگر $\mathsf{S} ::= \mathsf{if}(\mathsf{B})\mathsf{S}_t$ باشد، ردهای پیشوندی متناظر با اجرای این دستور را به شکل مجموعه‌ی زیر تعریف می کنیم:

$$\mathcal{S}^*[\![\mathsf{S}]\!] = \{\langle at[\![\mathsf{S}]\!], \rho\rangle | \rho \in \mathbb{EV}\} \cup \{\langle at[\![\mathsf{S}]\!], \rho\rangle\langle aft[\![\mathsf{S}]\!], \rho\rangle | \mathcal{B}[\![\mathsf{B}]\!]\rho = False\}$$

$$\cup\{\langle at[\![\mathsf{S}]\!], \rho\rangle\langle at[\![\mathsf{S_t}]\!], \rho\rangle\pi | \mathcal{B}[\![\mathsf{B}]\!]\rho = True \wedge \langle at[\![\mathsf{S_t}]\!], \rho\rangle\pi \in \mathcal{S}[\![\mathsf{S_t}]\!]\}$$

اگر $\mathsf{S} ::= \mathsf{if(B)S_t else S_f}$ باشد، ردهای پیشوندی متناظر با اجرای این دستور را به شکل مجموعه‌ی زیر تعریف می کنیم:

$$\mathcal{S}[\![\mathsf{S}]\!] = \{\langle at[\![\mathsf{S}]\!], \rho\rangle | \rho \in \mathbb{EV}\}$$

$$\cup\{\langle at[\![\mathsf{S}]\!], \rho\rangle\langle at[\![\mathsf{S_f}]\!], \rho\rangle\pi | \mathcal{B}[\![\mathsf{B}]\!]\rho = False \wedge \langle at[\![\mathsf{S_f}]\!], \rho\rangle\pi \in \mathcal{S}[\![\mathsf{S_f}]\!]\}$$
$$\cup\{\langle at[\![\mathsf{S}]\!], \rho\rangle\langle at[\![\mathsf{S_t}]\!], \rho\rangle\pi | \mathcal{B}[\![\mathsf{B}]\!]\rho = True \wedge \langle at[\![\mathsf{S_t}]\!], \rho\rangle\pi \in \mathcal{S}[\![\mathsf{S_t}]\!]\}$$

اگر $\mathsf{Sl} ::= \epsilon$ باشد، ردهای پیشوندی متناظر با اجرای این دستور را به شکل مجموعه‌ی زیر تعریف می کنیم:

$$\mathcal{S}[\![\mathsf{Sl}]\!] = \{\langle at[\![\mathsf{Sl}]\!], \rho\rangle | \rho \in \mathbb{EV}\}$$

اگر $\mathsf{Sl} ::= \mathsf{Sl'}\ \ \mathsf{S}$ باشد، ردهای پیشوندی متناظر با اجرای این دستور را به شکل مجموعه‌ی زیر تعریف می کنیم:

$$\mathcal{S}[\![\mathsf{Sl}]\!] = \mathcal{S}[\![\mathsf{Sl'}]\!] \cup (\mathcal{S}[\![\mathsf{Sl'}]\!] \rtimes\!\!\!\ltimes \mathcal{S}[\![\mathsf{S}]\!])$$

اگر $\mathsf{S} ::= \mathsf{while(B)S_b}$ باشد، ماجرا نسبت به حالات قبل اندکی پیچیده‌تر می‌شود. تابعی به اسم $\mathcal{F}$ را تعریف خواهیم‌کرد که در حقیت دو ورودی دارد. ورودی اول آن یک دستور حلقه است و ورودی دوم آن یک مجموعه. به عبارتی دیگر می‌توانیم بگوییم به ازای هر حلقه یک تابع $\mathcal{F}$ جداگانه تعریف می‌شود که مجموعه‌ای از ردهای پیشوندی را می گیرد و مجموعه‌ای دیگر از همین موجودات را بازمی‌گرداند. کاری که این تابع قرار است انجام دهد این است که انگار یک دور دستورات داخل حلقه را اجرا می کند و دنباله‌هایی جدید را از دنباله‌های قبلی می‌سازد. معنای یک حلقه را کوچکترین نقطه ثابت این تابع در نظر می‌گیریم. در ادامه تعریف $\mathcal{F}$ آمده. با دیدن تعریف می توان به دلیل این کار پی‌برد. آن نقطه‌ای که دیگر $\mathcal{F}$ روی آن اثر نمی‌کند یا حالتی است که در آن دیگر شرط حلقه برقرار نیست و اصولا قرار نیست دستورات داخل حلقه اجرا شوند که طبق تعریف $\mathcal{F}$ می‌توانیم ببینیم که $\mathcal{F}$ در این حالت چیزی به ردهای پیشوندی اضافه نمی‌کند. یا اینکه حلقه به دستور ;break خورده که در آن صورت وضعیتی به ته ردهای پیشوندی اضافه می‌شود که برچسبش خارج از مجموعه برچسب دستورات حلقه است و همین اضافه کردن هر چیزی را به ته ردهای پیشوندی موجود، توسط $\mathcal{F}$ غیرممکن می‌کند. بنابراین نقطه ثابت مفهوم مناسبی است برای اینکه از آن در تعریف صوری معنای حلقه استفاده کنیم. علت اینکه کوچکترین نقطه ثابت

را به عنوان معنای حلقه در نظر می‌گیریم هم این است که مطمئن هستیم کوچکترین نقطه ثابت، هر رد پیشوندی ای را در خود داشته باشد به معنای اجرای برنامه مرتبط است. برای درک بهتر این نکته می‌توان به این نکته توجه کرد که با اضافه کردن وضعیت‌هایی کاملا بی‌ربط به اجرای برنامه به ته ردهای پیشوندی، که صرفا برچسب متفاوتی با آخرین وضعیت هر رد پیشوندی دارند، نقطه ثابت جدیدی ساخته ایم. پس اگر خودمان را محدود به انتخاب کوچکترین نقطه ثابت نکنیم، به توصیفات صوری خوبی از برنامه‌ها دست پیدا نخواهیم‌کرد. در مورد نقطه ثابت تنها این نکته باقی می‌ماند که اصلا از کجا می‌دانیم که چنین نقطه ثابتی وجود دارد که در این صورت باید گفت مجموعه‌هایی که از ردهای پیشوندی تشکیل می‌شوند با عملگر زیرمجموعه بودن یک مشبکه را تشکیل می‌دهند و بنا به قضیه تارسکی[۲۳] برای چنین موجودی نقطه ثابت وجود دارد. تعاریف موجوداتی که درموردشان صحبت کردیم به این شکل است:

$$\mathcal{S}[\![\mathsf{S}]\!] = lfp^{\subseteq}\, \mathcal{F}[\![\mathsf{S}]\!]$$
$$\mathcal{F}[\![\mathsf{S}]\!]X = \{\langle at[\![\mathsf{S}]\!], \rho\rangle | \rho \in \mathbb{EV}\} \cup$$
$$\{\pi_2\langle l, \rho\rangle\langle aft[\![\mathsf{S}]\!], \rho\rangle | \pi_2\langle l, \rho\rangle \in X \wedge \mathcal{B}[\![\mathsf{B}]\!]\rho = False \wedge l = at[\![\mathsf{S}]\!]\} \cup$$
$$\{\pi_2\langle l, \rho\rangle\langle at[\![\mathsf{S_b}]\!], \rho\rangle\pi_3 | \pi_2\langle l, \rho\rangle \in X \wedge \mathcal{B}[\![\mathsf{B}]\!]\rho = True \wedge$$
$$\langle at[\![\mathsf{S_b}]\!], \rho\rangle\pi_3 \in \mathcal{S}[\![\mathsf{S_b}]\!] \wedge l = at[\![\mathsf{S}]\!]\}$$

اگر $\mathsf{S} ::= ;$ باشد، ردهای پیشوندی متناظر با اجرای این دستور را به شکل مجموعه‌ی زیر تعریف می کنیم:

$$\mathcal{S}[\![\mathsf{S}]\!] = \{\langle at[\![\mathsf{S}]\!], \rho\rangle | \rho \in \mathbb{EV}\} \cup \{\langle at[\![\mathsf{S}]\!], \rho\rangle\langle aft[\![\mathsf{S}]\!], \rho\rangle | \rho \in \mathbb{EV}\}$$

اگر $\mathsf{S} ::= \{\mathsf{Sl}\}$ باشد، ردهای پیشوندی متناظر با اجرای این دستور را به شکل مجموعه‌ی زیر تعریف می کنیم:

$$\mathcal{S}[\![\mathsf{S}]\!] = \mathcal{S}[\![\mathsf{Sl}]\!]$$

فصل ۳

صوری‌گری جدید برای روش وارسی مدل

۱.۳ ویژگی‌های معنایی برنامه‌ها

تا به اینجای کار یک زبان آورده‌ایم و برای آن معنا تعریف کرده‌ایم. در این فصل می‌خواهیم در مورد ویژگی‌های برنامه‌هایی که در این زبان نوشته می‌شوند با توجه به معنای صوری‌ای که تعریف کرده‌ایم، صحبت کنیم. دقت شود که برای برنامه‌هایی که در یک زبان برنامه‌نویسی نوشته می‌شوند می‌توان به اشکال مختلفی ویژگی تعریف کرد؛ مثلا ویژگی‌های نحوی، مثل اینکه طول برنامه چند خط است یا هر کاراکتر چند بار به کار رفته، یا ویژگی‌های محاسباتی، مثل بررسی سرعت برنامه یا میزان استفاده‌ی آن از حافظه که عموما در نظریه الگوریتم و پیچیدگی محاسبات بررسی می‌شود. منظور ما در اینجا از تعریف ویژگی، متناسب است با معناشناسی‌ای که برای برنامه‌هایمان تعریف کرده‌ایم. معناشناسی‌ای که تعریف کرده‌ایم درواقع سیر محاسباتی برنامه را توصیف می‌کند و ما می‌خواهیم ویژگی‌ها را با توجه به این موصوع تعریف کنیم. در این صورت می‌توانیم صحت عملکرد برنامه‌ها را با توجه به صادق بودن ویژگی‌هایی که در مورد آن‌ها تعریف شده بفهمیم.
ابتدا به تعریف ویژگی‌ها می‌پردازیم، سپس به سراغ تعریف یک نوع عبارت منظم می‌رویم که از آن برای بیان ویژگی‌ها استفاده می‌شود.

۱.۱.۳ ویژگی‌های معنایی

همان‌طور که در بخش قبلی دیدیم، معنای هر برنامه با یک مجموعه‌ی S^*S مشخص می‌شود. وقتی می‌خواهیم ویژگی‌هایی را برای موجوداتی که به کمک مجموعه‌ها تعریف شده اند بیان کنیم، اینکه ویژگی‌ها را هم با مجموعه‌ها بیان کنیم کار معقولی به نظر می‌رسد. مثل اینکه بخواهیم ویژگی زوج بودن را در مورد اعداد طبیعی بیان کنیم. می توانیم مجموعه‌ی $\mathbb{E}$ را به عنوان مجموعه‌ی همه‌ی اعداد زوج در نظر بگیریم و اینکه یک عدد زوج هست یا نه را عضویتش در مجموعه‌ی $\mathbb{E}$ تعریف

کنیم. پس یعنی در مورد اعداد طبیعی قرار است هر ویژگی به شکل زیرمجموعه‌ای از تمام این اعداد در نظر گرفته شود. یعنی هر عضو $P(\mathbb{N})$ بنا به تعریف ما یک ویژگی از اعداد طبیعی است.
در مورد برنامه‌ها نیز قرار است همین رویه را پیش بگیریم. تابع S^* از نوع $\mathbb{P} \to P(\mathfrak{S}^+)$ است.
یعنی یک برنامه را در ورودی می‌گیرد و یک مجموعه از ردهای پیشوندی را باز می‌گرداند. پس می‌توانیم هر ویژگی را به عنوان زیر مجموعه‌ای از $P(\mathfrak{S}^+)$ تعریف کنیم، به عبارت دیگر عضوی از $P(P(\mathfrak{S}^+))$.

۲.۱.۳ عبارات منظم

در اینجا توصیف ویژگی‌ها برای هر برنامه باید یک چارچوب داشته باشد. در صورت قدیمی روش وارسی مدل ما از منطق های زمانی برای بیان ویژگی‌ها به صورت صوری استفاده می‌کردیم و این احتیاج به یک زبان برای صوری کردن کامل کار را، که رسیدن به بیان مسئله‌ی وارسی مدل است، به ما نشان می‌دهد. در اینجا ما با داستان دیگری هم رو به رو هستیم و آن این است که از آنجایی که با مجموعه‌ها سر و کار داریم و مجموعه‌ها چندان موجودات ساختنی‌ای نیستند(برخلاف مدل کریپکی)، بهتر است یک موجود ساختنی مثل یک زبان صوری برای بیان آن‌ها داشته باشیم. در این فصل قصد داریم یک نوع عبارت منظم را برای این منظور تعریف کنیم. پیشتر به نکته‌ی دیگری در مورد استفاده از عبارات منظم، که متداول‌تر بودن بین جامعه‌ی برنامه نویسان است، صحبت کردیم. ابتدا زبان این عبارت منظم را تعریف می‌کنیم، سپس به سراغ معناشناسی آن می‌رویم.

۳.۱.۳ زبان عبارات منظم

فرق عمده‌ای که زبان عبارات منظم ما با عبارات منظم کلاسیک دارد در کاراکترهاست. کاراکترها در زبان کلاسیک موجوداتی اتمی بودند، اما در اینجا ساختار دارند. در اینجا به جای هر کاراکتر یک زوج متشکل از مجموعه‌ی L و عبارت بولی B تشکیل شده‌اند که این زوج را به شکل $\mathsf{L} : \mathsf{B}$ در زبانمان نمایش می‌دهیم.
زبان ما به شکل BNF زیر است:

تعریف ۱.۳.

$$\mathsf{L} \in P(\mathbb{L})$$

$$\mathsf{x}, \mathsf{y}, \ldots \in \mathbb{X}$$

$$\underline{\mathsf{x}}, \underline{\mathsf{y}}, \ldots \in \underline{\mathbb{X}}$$

$$\mathsf{B} \in \mathbb{B}$$

$$\mathsf{R} \in \mathbb{R}$$

$$
\begin{aligned}
\mathsf{R} ::= \quad & \varepsilon \\
\mid \ & \mathsf{L} : \mathsf{B} \\
\mid \ & \mathsf{R}_1\mathsf{R}_2 \quad (or\ \mathsf{R}_1 \bullet \mathsf{R}_2) \\
\mid \ & \mathsf{R}_1 \ \mid \ \mathsf{R}_2 \quad (or\ \mathsf{R}_1 + \mathsf{R}_2) \\
\mid \ & \mathsf{R}_1^* \\
\mid \ & \mathsf{R}_1^+ \\
\mid \ & (\mathsf{R}_1)
\end{aligned}
$$

همان طور که قابل مشاهده است در اینجا عملگرهای دوتایی چسباندن ($\bullet$) و انتخاب ($|$) را داریم، به همراه عملگرهای یگانی * و $^+$. در ادامه خواهیم دید که در فرازبان معنی عملگر یگانی $^+$ به وسیله‌ی عملگر یگانی دیگر قابل بیان است، هرچند که در زبانمان هم برای سهولت کار از بیان این عملگر اجتناب نشده. توجه شود که پرانتزها هم جزئی از زبان قرار داده شده‌اند.

همین‌طور در اینجا می‌خواهیم از تعدادی عبارات مخفف که در ادامه کارمان را راحت‌تر می‌کنند صحبت کنیم. منظور از زوج B : ? همان $\mathbb{L}$:B است. عبارت B : l به جای عبارت B : $\{l\}$ به کار می‌رود و منظور از عبارت B :l¬ نیز عبارت B : $\mathbb{L} \setminus \{l\}$ است.

با یک نگاه به دستور این زبان یک نکته‌ی چشمگیر برای ما، با توجه به موجوداتی که در بخش قبل تعریف کردیم، با نگاه به قواعد این زبان می‌تواند وجود یک مجموعه‌ی $\underline{\mathbb{X}}$ در کنار $\mathbb{X}$ که از قبل داشتیم باشد. قرار است به ازای هر $\mathsf{x} \in \mathbb{X}$ یک $\underline{\mathsf{x}} \in \underline{\mathbb{X}}$ داشته‌باشیم. منظور از $\underline{\mathsf{x}}$ مقدار متغیر x در ابتدای هر برنامه است. این یعنی تابع $\underline{\rho} : \underline{\mathbb{X}} \to \mathbb{V}$ که $\mathbb{V}$ مجموعه‌ی مقادیر متغیرهاست(در بخش قبل به این اشاره نشد اما خود ρ هایی که در بخش قبل داشتیم هم از نوع $\mathbb{X} \to \mathbb{V}$ بود. با توجه به زبانمان و توضیحاتی که در گذشته دادیم، می‌توان در نظر گرفت که در اینجا $\mathbb{V}$ همان اعداد صحیح است). همان‌طور که پیش‌تر گفتیم برای اشاره به یک تابع ρ از کلمه‌ی "محیط" استفاده می‌شود. به همین منوال در ادامه برای اشاره به $\underline{\rho}$ از "محیط اولیه" استفاده می‌کنیم. برای اشاره به مجموعه‌ی همه‌ی محیط‌های اولیه هم از نماد $\underline{\mathbb{EV}}$ استفاده می‌کنیم. بقیه‌ی موجودات از جمله برچسب‌ها و عبارات بولی را هم که قبلا داشتیم.

در ادامه به بیان صوری معنای زبان بیان شده می‌پردازیم.

۴.۱.۳ معناشناسی عبارات منظم

معنای عبارات منظم را با استفاده از تابع S^r نشان می‌دهیم. این تابع به این شکل تعریف می‌شود که در ورودی یک عبارت منظم R را می‌گیرد، سپس یک مجموعه از زوج مرتب‌های(یا همان‌طور که

پیش‌تر نام‌گذاری کردیم "وضعیت‌ها"ی) $\langle \underline{\rho}, \pi \rangle$ را که $\pi \in \mathbb{S}^*$ و $\underline{\rho} \in \underline{\mathbb{EV}}$ باز می‌گرداند. بنابراین این تابع از نوع $\mathbb{R} \to P(\underline{\mathbb{EV}} \times \mathbb{S}^*)$ است. همین‌طور دقت شود که تا به حال از $\mathbb{S}^*$ صحبتی نکرده بودیم و فقط $\mathbb{S}^+$ را معرفی کرده بودیم. $\mathbb{S}^*$ نیز برابر است با $\mathbb{S}^+ \cup \{\epsilon\}$ (به لحاظ معنایی همان عملگر * است که در زبان عبارات منظمهم هست، مشهور به ستاره‌ی کلینی).

تعریف استقرایی تابع $\mathcal{S}^r$ به شکل زیر است:

تعریف ۲.۳. تابع $\mathcal{S}^r : \mathbb{R} \to P(\underline{\mathbb{EV}} \times \mathbb{S}^*)$ به صورت استقرایی روی ساختار عبارت منظم R به صورت زیر تعریف می‌شود:

$$\mathcal{S}^r[\![\varepsilon]\!] = \{\langle \underline{\rho}, \epsilon \rangle | \underline{\rho} \in \underline{\mathbb{EV}}\}$$

[یعنی معنای عبارت منظم ε مجموعه‌ای شامل زوج مرتب‌هایی از محیط‌های اولیه‌ی مختلف در کنار رد پیشوندی تهی استفاده می‌کند.]

$$\mathcal{S}^r[\![\mathsf{L} : \mathsf{B}]\!] = \{\langle \underline{\rho}, \langle l, \rho \rangle \rangle | l \in \mathsf{L} \wedge \mathcal{B}[\![\mathsf{B}]\!]\underline{\rho}, \rho\}$$

[این یعنی معنای عبارت $\mathcal{S}^r[\![\mathsf{L} : \mathsf{B}]\!]$ زوج مرتب‌هایی هستند که عضو اول آن‌ها محیط‌های اولیه مختلف هستند(مانند مورد قبلی و البته در موارد آتی!) و عضو دوم آن‌ها ردهای پیشوندی تک‌عضوی $\langle l, \rho \rangle$ هستند که در آن‌ها برچسب l باید در L که مجموعه‌ای از برچسب‌هاست حضور داشته باشد و عبارت بولی B باید درباره‌ی محیط اولیه $\underline{\rho}$ و محیط ρ برقرار باشد. حتما متوجه این نکته شدید که $\mathcal{B}$ در اینجا به جای اینکه از نوع $\mathbb{EV} \to \mathbb{BOOL}$ باشد،همان‌طور که قبلا تعریف کردیم، از نوع $\underline{\mathbb{EV}} \to \mathbb{EV} \to \mathbb{BOOL}$ است.(منظور از $\mathbb{BOOL}$ همان مجموعه‌ی $\{True, False\}$ است.) در اینجا $\mathcal{A}$ و $\mathcal{B}$ را در ادامه با نوع‌های متفاوت دوباره تعریف خواهیم کرد، که البته فرق اساسی‌ای با تعریف قبلی ندارد و صرفا گسترشی ساده از آن است.]

$$\mathcal{S}^r[\![\mathsf{R}_1\mathsf{R}_2]\!] = \mathcal{S}^r[\![\mathsf{R}_1]\!] \bowtie \mathcal{S}^r[\![\mathsf{R}_2]\!]$$

به‌طوری که در آن برای هر دو مجموعه‌ی $\mathcal{S}$ و $\mathcal{S}'$ از ردهای پیشوندی:

$$\mathcal{S} \bowtie \mathcal{S}' = \{\langle \underline{\rho}, \pi\pi' \rangle | \langle \underline{\rho}, \pi \rangle \in \mathcal{S} \wedge \langle \underline{\rho}, \pi' \rangle \in \mathcal{S}'\}$$

[این یعنی اگر یک عبارت منظم داشته باشیم که از چسباندن R_1 و R_2 به هم ساخته شده باشد، آنگاه معنای این عبارت منظم با چسباندن ردهای پیشوندی موجود در مولفه‌ی دوم زوج مرتب‌هایی که اعضای مجموعه‌ی معنای این دو عبارت منظم هستند و گذاشتن این رد پیشوندی‌های حاصل از چسباندنْ در معنای عبارت منظم جدید تعریف می‌شود. همین‌طور که می‌بینید یک عملگر چسباندن برای دو مجموعه از این زوج‌های $\langle \underline{\rho}, \pi \rangle$ تعریف شده و در تعریف $\mathcal{S}^r[\![\mathsf{R}_1\mathsf{R}_2]\!]$ از آن کمک گرفته شده.

تا این تکه از تعریف معنای عبارت منظم که رسیده‌ایم، تا حدی به دستیابی به درکی شهودی از اینکه

به چه نحوی قرار است عبارات منظم راهی برای توصیف ویژگی در مورد برنامه‌ها باشد نزدیک‌تر شده‌ایم. همان‌طور که در مورد قبل دیدیم هر زوج $\mathsf{L}:\mathsf{B}$ دقیقا به یک وضعیتْ داخل یک رد پیشوندی اشاره می‌کند. انگار که قرار است این زوج‌ها موازی با وضعیت‌ها در ردهای پیشوندی موجود در معنای یک برنامه جلو روند و منطبق باشند تا وارسی مدل انجام شود. درک این موضوع اولین قدم ماست در دیدن عصاره‌ی روش وارسی مدل در ادبیاتی که از اول این فصلْ عَلَم کرده‌ایم.]

$$\mathcal{S}^r[\![\mathsf{R}_1 \mid \mathsf{R}_2]\!] = \mathcal{S}^r[\![R_1]\!] \cup \mathcal{S}^r[\![R_2]\!]$$

[این موردْ معنای اعمال عملگر انتخاب روی دو عبارت منظم را توصیف می‌کند. معنای اعمال این عملگر به‌سادگی به صورت اجتماع معنای هر دو عبارت منظم تعریف شده.]

$$\mathcal{S}^r[\![\mathsf{R}]\!]^0 = \mathcal{S}^r[\![\varepsilon]\!]$$
$$\mathcal{S}^r[\![\mathsf{R}]\!]^{n+1} = \mathcal{S}^r[\![\mathsf{R}]\!]^n \bowtie \mathcal{S}^r[\![\mathsf{R}]\!]$$

[دو عبارت اخیر برای توصیف معنای عملگرهای * و $^+$ تعریف شده‌اند. عملگر $\bowtie$ و معنای $\mathcal{S}^r[\![\varepsilon]\!]$ را هم که قبلا تعریف کرده بودیم و 0 و n و $n+1$ هم اعداد طبیعی‌اند و $+$ لاجرم همان جمع اعداد طبیعی است.]

$$\mathcal{S}^r[\![\mathsf{R}^*]\!] = \bigcup_{n\in\mathbb{N}} \mathcal{S}^r[\![\mathsf{R}^n]\!]$$

$$\mathcal{S}^r[\![\mathsf{R}^+]\!] = \bigcup_{n\in\mathbb{N}\setminus\{0\}} \mathcal{S}^r[\![\mathsf{R}^n]\!]$$

[این دو عبارت هم تعریف معنای خود دو عملگر * و $^+$ هستند. منظور از $\mathbb{N}$ مجموعه‌ی اعداد طبیعی است. همان‌طور که قبل‌تر هم اشاره شد $^+$ را می‌توان در فرازبان با * تعریف کرد. اضافه می‌کنیم که خود * را هم در فرازبان می‌توان با عملگر انتخاب تعریف کرد و در اینجا می‌توان این نکته را هم دید.]

$$\mathcal{S}^r[\![(\mathsf{B})]\!] = \mathcal{S}^r[\![\mathsf{B}]\!]$$

[این تکه از تعریف هم صرفا بیان می‌کند که پرانتزها تاثیری در معنای عبارات منظم ندارند که کاملا قابل انتظار است چرا که وجود پرانتز قرار است در صرفا در خواص نحوی زبان اثر بگذارد.]

تعریف معنای عبارات منظم در اینجا تمام می‌شود اما همان‌گونه که در لابه‌لای تعاریف گفتیم، احتیاج داریم که $\mathcal{A}$ و $\mathcal{B}$ را از نو تعریف کنیم:

تعریف ۳.۳. توابع $\mathcal{A}:\mathbb{A}\to\underline{\mathbb{EV}}\to\mathbb{EV}\to\mathbb{V}$ و $\mathcal{B}:\mathbb{B}\to\underline{\mathbb{EV}}\to\mathbb{EV}\to\mathbb{BOOL}$ به شکل استقرایی به ترتیب روی ساختارهای $\mathsf{A}\in\mathbb{A}$ و $\mathsf{B}\in\mathbb{B}$ به شکل زیر تعریف می‌شوند:

$$\mathcal{A}[\![1]\!]\underline{\rho},\rho = 1$$
$$\mathcal{A}[\![\underline{\mathsf{x}}]\!]\underline{\rho},\rho = \underline{\rho}(\mathsf{x})$$
$$\mathcal{A}[\![\mathsf{x}]\!]\underline{\rho},\rho = \rho(\mathsf{x})$$
$$\mathcal{A}[\![\mathsf{A}_1 - \mathsf{A}_2]\!]\underline{\rho},\rho = \mathcal{A}[\![\mathsf{A}_1]\!]\underline{\rho},\rho - \mathcal{A}[\![\mathsf{A}_2]\!]\underline{\rho},\rho$$
$$\mathcal{B}[\![\mathsf{A}_1 < \mathsf{A}_2]\!]\underline{\rho},\rho = \mathcal{A}[\![\mathsf{A}_1]\!]\underline{\rho},\rho \ < \ \mathcal{A}[\![\mathsf{A}_2]\!]\underline{\rho},\rho$$
$$\mathcal{B}[\![\mathsf{B}_1 \ \mathsf{nand}\ \mathsf{B}_2]\!]\underline{\rho},\rho = \mathcal{B}[\![\mathsf{B}_1]\!]\underline{\rho},\rho \uparrow \mathcal{B}[\![\mathsf{B}_2]\!]\underline{\rho},\rho$$

به‌راحتی قابل مشاهده است که تعاریف جدید تا حد خوبی به تعاریف قبلی شبیه هستند و فرق عمده صرفا وارد شدن $\underline{\rho}$ است.

حال که معناشناسی عبارات منظم را داریم، به طور مختصر به مقایسه‌ی عبارات منظمی که در این بحث تعریف کرده‌ایم و عبارات منظم کلاسیک در باقی نوشته‌ها و موضوعات می‌پردازیم. جبر کلاینی یک ساختار جبری است که تعمیمی است از عبارات منظم معرفی شده در [۱۵]. سر و کله‌ی عبارات منظم در قسمت‌های مختلفی از علوم کامپیوتر پیدا شده، با تعاریفی نابرابر. جبر کلاینی قرار بوده که تعمیمی باشد که این تعاریف نابرابر را در خود جای دهد. در [۱۶] آمده که برای جبر کلاینی هم تعاریف متفاوتی که با هم برابر نیستند، معرفی شده است. علاوه بر این، این مقاله به بررسی این تعاریف و ارتباطشان با یکدیگر پرداخته است. همین طور، این مقاله خود با یک تعریف از جبر کلاینی شروع کرده. طبق این تعریف اگر عبارات منظم ما یک جبر کلاینی می‌بودند، می‌بایستی که برای هر عبارت منظم R می‌داشتیم $\mathcal{S}^r[\![\varepsilon]\!] = \mathcal{S}^r[\![\varepsilon\mathsf{R}]\!]$ چون ε نقش صفر عملگر انتخاب(که عملگر جمع جبر کلینی است) را دارد و یک جبر کلاینی برای صفر خاصیت جذب را به عنوان یک اصل دارد. اما در مورد عبارات منظمی که در اینجا تعریف کردیم داریم $\mathcal{S}^r[\![\varepsilon\mathsf{R}]\!] = \mathcal{S}^r[\![\mathsf{R}]\!]$. بیشتر از این به این بحث نمی‌پردازیم که بحث دامنه‌دار و منحرف کننده‌ایست. یک قضیه را در مورد عبارات منظم ارائه می‌دهیم که پخش پذیری عملگر انتخاب به عملگر چسباندن است و بعد به ادامه‌ی راه اصلیمان می‌پردازیم.

قضیه ۴.۳. برای عبارات منظم $\mathsf{R}, \mathsf{R}_1, \mathsf{R}_2$ داریم:

$$\mathcal{S}^r[\![\mathsf{R} \bullet (\mathsf{R}_1 + \mathsf{R}_2)]\!] = \mathcal{S}^r[\![(\mathsf{R} \bullet \mathsf{R}_1) + (\mathsf{R} \bullet \mathsf{R}_2)]\!]$$

اثبات.

$$\mathcal{S}^r[\![\mathsf{R} \bullet (\mathsf{R}_1 + \mathsf{R}_2)]\!] = \mathcal{S}^r[\![\mathsf{R}]\!] \bowtie \mathcal{S}^r[\![(\mathsf{R}_1 + \mathsf{R}_2)]\!]$$
$$= \mathcal{S}^r[\![\mathsf{R}]\!] \bowtie \mathcal{S}^r[\![\mathsf{R}_1 + \mathsf{R}_2]\!] = \mathcal{S}^r[\![\mathsf{R}]\!] \bowtie (\mathcal{S}^r[\![\mathsf{R}_1]\!] \cup \mathcal{S}^r[\![\mathsf{R}_2]\!])$$
$$= \{\langle\underline{\rho},\pi\pi'\rangle | \langle\underline{\rho},\pi\rangle \in \mathcal{S}^r[\![\mathsf{R}]\!] \wedge (\langle\underline{\rho},\pi'\rangle \in \mathcal{S}^r[\![\mathsf{R}_1]\!] \vee \langle\underline{\rho},\pi'\rangle \in \mathcal{S}^r[\![\mathsf{R}_2]\!])\}$$

$$= \{\langle \underline{\rho}, \pi\pi' \rangle | (\langle \underline{\rho}, \pi \rangle \in \mathcal{S}^r[\![\mathsf{R}]\!] \wedge \langle \underline{\rho}, \pi' \rangle \in \mathcal{S}^r[\![\mathsf{R}_1]\!]) \vee (\langle \underline{\rho}, \pi \rangle \in \mathcal{S}^r[\![\mathsf{R}]\!] \wedge \langle \underline{\rho}, \pi' \rangle \in \mathcal{S}^r[\![\mathsf{R}_2]\!]\}$$
$$= \{\langle \underline{\rho}, \pi\pi' \rangle | \langle \underline{\rho}, \pi \rangle \in \mathcal{S}^r[\![\mathsf{R}]\!] \wedge \langle \underline{\rho}, \pi' \rangle \in \mathcal{S}^r[\![\mathsf{R}_1]\!]\} \cup \{\langle \underline{\rho}, \pi\pi' \rangle | \langle \underline{\rho}, \pi \rangle \in \mathcal{S}^r[\![\mathsf{R}]\!] \wedge \langle \underline{\rho}, \pi' \rangle \in \mathcal{S}^r[\![\mathsf{R}_2]\!]\}$$
$$= (\mathcal{S}^r[\![\mathsf{R}]\!] \bowtie \mathcal{S}^r[\![\mathsf{R}_1]\!]) \cup (\mathcal{S}^r[\![\mathsf{R}]\!] \bowtie \mathcal{S}^r[\![\mathsf{R}_2]\!]) = \mathcal{S}^r[\![\mathsf{R} \bullet \mathsf{R}_1]\!] \cup \mathcal{S}^r[\![\mathsf{R} \bullet \mathsf{R}_2]\!]$$
$$= \mathcal{S}^r[\![(\mathsf{R} \bullet \mathsf{R}_1) + (\mathsf{R} \bullet \mathsf{R}_2)]\!]$$

□

تا اینجای کار بیشتر مفاهیمی که برای بیان صورت جدید مسئله‌ی وارسی مدل احتیاج داریم را بیان کرده‌ایم.

۵.۱.۳ گونه‌های مختلف زبان عبارات منظم

به عنوان قسمت آخر این بخش، گونه‌های مختلفی از زبان عبارات منظم را بیان می‌کنیم که هر کدام در واقع زیرمجموعه‌هایی از کل عبارات زبانی که توصیف کرده‌ایم را توصیف می‌کنند. بعضی از آن‌ها را در همین فصل برای هدف نهایی این فصل و بعضی دیگر را در فصل بعدی استفاده می‌کنیم.

اولین گونه‌ای که می‌خواهیم بیان کنیم، گونه‌ای است که در اعضای آن اصلا عبارت $\mathsf{L} : \mathsf{B}$ حضور ندارد و کل عبارت‌های زبان از ε ها تشکیل شده‌اند.

تعریف ۵.۳. (عبارت منظم تهی ـ $\mathbb{R}_\varepsilon$):

$$\mathsf{R} \in \mathbb{R}_\varepsilon$$

$$\mathsf{R} ::= \varepsilon \mid \mathsf{R}_1\mathsf{R}_2 \mid \mathsf{R}_1 + \mathsf{R}_2 \mid \mathsf{R}_1^* \mid \mathsf{R}_1^+ \mid (\mathsf{R}_1)$$

با توجه به بخش قبل متوجه هستیم که معنای همه‌ی این عبارت‌ها برابر $\{\langle \underline{\rho}, \epsilon \rangle\}$ خواهد بود.
گونه‌ی بعدی عبارت منظم ناتهی است.

تعریف ۶.۳. (عبارت منظم ناتهی ـ $\mathbb{R}^+$):

$$\mathsf{R} \in \mathbb{R}^+$$

$$\mathsf{R} ::= \mathsf{L} : \mathsf{B} \mid \varepsilon\mathsf{R}_2 \mid \mathsf{R}_1\varepsilon \mid \mathsf{R}_1\mathsf{R}_2 \mid \mathsf{R}_1 + \mathsf{R}_2 \mid \mathsf{R}_1^+ \mid (\mathsf{R}_1)$$

دلیل وجود $\varepsilon\mathsf{R}_2$ و $\mathsf{R}_1\varepsilon$ در تعریف این است که ممکن است معنای عبارتی با معنای عبارات عضو $\mathbb{R}_\varepsilon$ برابر نباشد(یعنی برابر $\langle \underline{\rho}, \epsilon \rangle$ نباشد)، اما در خود عبارت، ε حضور داشته باشد. با این تفاصیل می‌توان دید که دو مجموعه‌ی $\mathbb{R}_\varepsilon$ و $\mathbb{R}^+$ یک افراز برای مجموعه‌ی $\mathbb{R}$ هستند، براساس اینکه معنای هر عبارت در $\mathbb{R}$ برابر $\langle \underline{\rho}, \epsilon \rangle$ هست یا خیر. بنابراین شاید به نظر برسد که تعریف یکی از آن‌ها به طور ساختاری کافی بود، اما ممکن است درجایی احتیاج داشته باشیم که ساختاری استقرایی

روی هر یک از آن‌ها عَلَم کنیم یا اینکه در اثبات حکمی بخواهیم از استقرا روی یکی از این دو ساختار استفاده کنیم.

گونه‌ی آخر عبارات منظم ما نیز عبارات منظم بدون انتخاب است.

تعریف ۷.۳. (عبارت منظم بدون انتخاب ـ $\mathbb{R}^{\nmid}$):

$$\mathsf{R} \in \mathbb{R}^{\nmid}$$

$$\mathsf{R} ::= \varepsilon \mid \mathsf{L} : \mathsf{B} \mid \mathsf{R}_1\mathsf{R}_2 \mid \mathsf{R}_1{}^* \mid \mathsf{R}_1{}^+ \mid (\mathsf{R}_1)$$

۲.۳ صورت جدید مسئله‌ی وارسی مدل

بالاخره به هدف نهایی این فصل رسیدیم. می‌خواهیم صورت جدیدی از مسئله‌ی وارسی مدل را بیان کنیم.

پیش از ارائه‌ی تعریف وارسی مدل نیاز داریم تا عملگر بستار پیشوندی را برای یک مجموعه از ردهای پیشوندی معرفی کنیم.

تعریف ۸.۳. (بستار پیشوندی): اگر $\Pi \in P(\underline{\mathbb{EV}} \times \mathbb{S}^+)$ آنگاه بستار پیشوندی Πرا به صورت زیر تعریف می‌کنیم:

$$\mathsf{prefix}(\Pi) = \{\langle \underline{\rho}, \pi\rangle | \pi \in \mathbb{S}^+ \wedge \exists\, \pi' \in \mathbb{S}^* : \langle \underline{\rho}, \pi\pi'\rangle \in \Pi\}$$

برای درک بهتر مفهوم بستار پیشوندی به مثال زیر توجه شود.

مثال ۹.۳. اگر $\Pi = \{\langle \underline{\rho}, \langle l_1, \rho_1\rangle\langle l_2\rho_2\rangle\rangle\langle l_3, \rho_3\rangle\rangle, \langle \underline{\rho}, \langle l_1{}', \rho_1{}'\rangle\langle l_2{}'\rho_2{}'\rangle\rangle\}$ باشد(Π شامل دو عضو است) آنگاه:

$$\mathsf{prefix}(\Pi) = \{\langle \underline{\rho}, \langle l_1, \rho_1\rangle\rangle, \langle \underline{\rho}, \langle l_1, \rho_1\rangle\langle l_2\rho_2\rangle\rangle, \langle \underline{\rho}, \langle l_1, \rho_1\rangle\langle l_2\rho_2\rangle\langle l_3, \rho_3\rangle\rangle,$$
$$\langle \underline{\rho}, \langle l_1{}', \rho_1{}'\rangle\rangle, \langle \underline{\rho}, \langle l_1{}', \rho_1{}'\rangle\langle l_2{}'\rho_2{}'\rangle\rangle\}$$

که شامل ۵ عضو است.

حال به ارائه‌ی صورت جدیدمان از روش وارسی مدل می‌رسیم که هدف اصلی این اصل بود و با این تعریف فصل تمام می‌شود.

تعریف ۱۰.۳. (وارسی مدل): اگر $\mathsf{P} \in \mathbb{P}, \mathsf{R} \in \mathbb{R}^+, \underline{\rho} \in \underline{\mathbb{EV}}$ آنگاه:

$$\mathsf{P}, \underline{\rho} \models \mathsf{R} \Leftrightarrow (\{\underline{\rho}\} \times \mathcal{S}^*[\![\mathsf{P}]\!]) \subseteq \mathsf{prefix}(\mathcal{S}^r[\![\mathsf{R} \bullet (?:\ T)^*]\!])$$

این تعریف بیان می‌کند که برنامه‌ی P در صورتی که با محیط اولیه‌ی $\underline{\rho}$ اجرا شود، در صورتی خاصیتی که با عبارت منظم R بیان شده را دارد که معنای آن زیرمجموعه‌ی بستار پیشوندی معنای عبارت منظم $(? \,:\, ?) \bullet$ R $^{*}(T)$ باشد. توجه شود که محیط اولیه‌ای که برای برنامه‌ی مورد بررسی متصور هستیم صرفا به این منظور قرار داده شده که معناشناسی برنامه را بتوانیم با معنای عبارات منظم قابل قیاس کنیم. دلیل حضور محیط اولیه در معنای عبارات منظم نیز در صورت سوم روش وارسی مدل یعنی فصل ۴ مشخص می‌شود و در این صورت از روش وارسی مدل و صورت بعدی آن چندان نقشی ندارد.

در مورد نقش $^{*}(T \,:\, ?)$ و prefix این به تصمیم مبدع این روش بوده که این دو در تعریف روش وارسی مدل حضور داشته باشند. حضور این دو طبعا باعث می‌شود به ازای یک عبارت منظم R نسبت به این حالت که صرفا معنی برنامه زیرمجموعه‌ی معنی R باشد، برنامه‌های بیشتری باشند که خاصیت بیان شده با R را ارضا کنند، چون در این صورت مجموعه‌ی سمت راستی در رابطه‌ی زیرمجموعه بودن بزرگ‌تر می‌شود.

۳.۳ در مورد توقف پذیری

در این بخش نکته‌ای در مورد کار که به نظر نگارنده رسیده مطرح شده. اگر صحبت ما در اینجا درست باشد، این به این معنی خواهد بود که کل کاری که در حال توصیفش هستیم قابل پیاده سازی نیست!

بحث ما در اینجا در مورد توقف پذیری است. در [۶] در مورد توقف یک برنامه صحبتی به میان نیامده. یعنی حتی گفته نشده که در چه صورتی می‌توانیم بگوییم که یک برنامه متوقف شده است. یک تعریف صوری معقول که خودمان می‌توانیم برای این معنا بیاوریم این است:

تعریف ۳.۱۱ . (توقف پذیری:) برنامه‌ی P را به همراه اجرای اولیه $\underline{\rho}$ توقف پذیر می‌گوییم اگر و تنها اگر وجود داشته باشد $\pi \in \mathcal{S}^*[\![\mathsf{P}]\!]$ که (ρ محیط متناظر با محیط اولیه‌ی $\underline{\rho}$ است.):

$$\pi = \langle at[\![\mathsf{P}]\!], \rho\rangle \pi'$$

و اینکه $\langle aft[\![\mathsf{P}]\!], \rho'\rangle$ در π حضور داشته باشد. این اتفاق را با $\mathsf{P}, \underline{\rho} \downarrow$ نشان می‌دهیم. همین تعریف را برای لیست دستورات Sl یا دستور S هم صرفا با جایگذاری این‌ها با برنامه‌ی P داریم.

در این تعریف توقف پذیری صرفا برای یک محیط اولیه تعریف شده. در اینجا توقف پذیری به متناهی بودن ردهای پیشوندی موجود در برنامه ربط داده نشده. با توجه به معناشناسی‌ای که داریم، تعریف توقف پذیری به معنای وجود رد پیشوندی متناهی با محیط اولیه‌ی مورد بررسی در معنای برنامه که اصلا جور در نمی‌آید، چون معناشناسی ما خاصیت پیشوندی بودن را دارد و مطمئن هستیم در معنای هر برنامه‌ای حتما یک رد پیشوندی متناهی با محیط اولیه‌ی مورد بررسی وجود دارد.

اگر هم بخواهیم تعریف توقف پذیری را وجودنداشتن ردهای پیشوندی نامتناهی با محیط اولیه‌ی مورد بررسی در معنای برنامه در نظر بگیریم در ابتدا به نظر می‌آید که به تعریف قوی‌تری نسبت به آنچه ارائه دادیم رسیده‌ایم. ما در اینجا سعی داریم تعریفی را ارائه کنیم که برای حرف‌هایی که در [۶] زده شده تا حد امکان مشکل درست نکند، که اگر دیدیم با این وجود مشکل وجود دارد مطمئن باشیم که اشتباه در [۶] است و نه حرف ما. پس سعی از ارائه‌ی این تعریف که به نظر از تعریف ارائه شده با کار ناسازگارتر می‌آید اجتناب می‌کنیم(در ادامه به بیان ناسازگاری پراخته شده) اما در قضیه‌ی بعدی می‌بینیم که تعریفی که ارائه کردیم با همان که بگوییم در برنامه رد پیشوندی نامتناهی وجود ندارد معادل است.

قضیه ۱۲.۳. برای برنامه‌ی P و محیط اولیه‌ی $\underline{\rho}$ داریم $\mathsf{P}, \underline{\rho} \downarrow$ اگر و تنها اگر با فرض اینکه ρ محیط متناظر با محیط اولیه‌ی $\underline{\rho}$ است و

$$\forall \pi \in \mathbb{S}^{+} : \langle at[\![\mathsf{P}]\!], \rho\rangle\pi \in \mathcal{S}^{*}[\![\mathsf{P}]\!] \to \langle at[\![\mathsf{P}]\!], \rho\rangle\pi \in \mathbb{R}^{+}$$

اثبات. $(\Rightarrow)$ برای این قسمت باید ثابت کنیم که در معنای هر برنامه‌ای رد پیشوندی‌ای وجود دارد که با $\langle at[\![\mathsf{P}]\!], \rho\rangle$ شروع شده و به ازای یک محیط ρ' به $\langle aft[\![\mathsf{P}]\!], \rho'\rangle$ ختم شده. در این اثبات از تعریف برچسب‌ها که در ضمیمه‌ی [۶] آمده استفاده شده. داریم $\mathsf{P} = \mathsf{Sl}$ و $aft[\![\mathsf{P}]\!] = aft[\![\mathsf{Sl}]\!]$ حکم را با استقرا روی ساختار Sl ثابت می‌کنیم.

$$\blacktriangleright\ \mathsf{Sl} = \ni :$$

داریم:

$$\mathcal{S}^{*}[\![\ni]\!] = \{\langle at[\![\ni]\!], \dot{\rho}\rangle | \dot{\rho} \in \mathbb{EV}\}$$

و طبق تعریف برچسب‌ها داریم:

$$at[\![\ni]\!] = aft[\![\ni]\!]$$

پس حکم برقرار است.

$$\blacktriangleright\ \mathsf{Sl} = \mathsf{Sl}'\ \mathsf{S} :$$

اینکه در معنای Sl دنباله‌ای شامل $\langle aft[\![\mathsf{Sl}]\!], \rho'\rangle$ وجود داشته باشد، به با توجه به تعاریفی که داریم به این وابسته است که در معنای S دنباله‌ای شامل $\langle aft[\![\mathsf{S}]\!], \rho'\rangle$ وجود داشته باشد. برای اینکه این را ثابت کنیم هم باید همین حکم را روی ساختار S ثابت کنیم که در واقع بخش اصلی اثبات این سمت قضیه است.

$$\blacktriangleright\blacktriangleright\ \mathsf{S}\ =\ \mathsf{x} \doteq \mathsf{A}; :$$

در این حالت با توجه به تعریف معنای S که قبل‌تر ارائه شد، دنباله‌ی

$$\langle at[\![\mathsf{S}]\!], \rho\rangle\langle aft[\![\mathsf{S}]\!], \rho[\mathsf{x} \leftarrow \mathcal{A}[\![\mathsf{A}]\!]\rho]\rangle$$

در معنای دستور به ازای هر ρ وجود دارد که خب در هر صورت این شامل محیط متناظر با $\underline{\rho}$ هم می‌شود .

$\blacktriangleright\blacktriangleright$ $\mathsf{S} = ;$:

با توجه به معنای این دستوردنباله‌ی زیر در معنای این دستور وجود دارد.

$$\langle at[\![\mathsf{S}]\!], \rho\rangle\langle aft[\![\mathsf{S}]\!], \rho\rangle$$

$\blacktriangleright\blacktriangleright$ $\mathsf{S} =$ if (B) $\mathsf{S_t}$:

در صورتی که $\mathcal{B}[\![\mathsf{B}]\!]\rho = T$ دنباله‌ی

$$\langle at[\![\mathsf{S}]\!], \rho\rangle\langle at[\![\mathsf{S_t}]\!], \rho\rangle\pi$$

در مجموعه‌ی معنای این دستور حضور دارد در حالیکه $\langle at[\![\mathsf{S_t}]\!], \rho\rangle\pi$ داخل معنای $\mathsf{S_t}$ است و طبق فرض استقرا می‌دانیم که برچسب آخرین موقعیت π برابر است با $aft[\![\mathsf{S_t}]\!]$ که طبق تعاریف مربوط به برچسب‌ها $aft[\![\mathsf{S_t}]\!] = aft[\![\mathsf{S}]\!]$. در صورتی که معنای عبارت بولی غلط باشد هم دنباله‌ی زیر در معنای دستور طبق تعریف موجود است.

$$\langle at[\![\mathsf{S}]\!], \rho\rangle\langle aft[\![\mathsf{S}]\!], \rho\rangle$$

$\blacktriangleright\blacktriangleright$ $\mathsf{S} =$ if (B) $\mathsf{S_t}$ else $\mathsf{S_f}$:

مانند حالت قبل است منتها با این تفاوت که در صورتی که معنای عبارت بولی غلط باشد دنباله‌ی زیر در معنای دستور حضور دارد:

$$\langle at[\![\mathsf{S}]\!], \rho\rangle\langle at[\![\mathsf{S_f}]\!], \rho\rangle\pi$$

و تساوی $aft[\![\mathsf{S_t}]\!] = aft[\![\mathsf{S}]\!] = aft[\![\mathsf{S_f}]\!]$ هم طبق تعریف برچسب‌ها برقرار است.

$\blacktriangleright\blacktriangleright$ $\mathsf{S} =$ while (B) $\mathsf{S_t}$:

در اثبات این سمت قضیه این حالت پیچیده ترین حالت است و در واقع تنها حالتی است که در اثبات آن به فرض قضیه احتیاج داریم! همان طور که پیشتر گفتیم معنای حلقه با استفاده از یک تابع تعریف می‌شود. معنای حلقه کوچکترین نقطه ثابت این تابع است، در حالیکه انگار این تابع وقتی روی یک مجموعه از ردهای پیشوندی اعمال شود، تاثیرات یک بار اجرای دستورات درون حلقه را روی ردهای پیشوندی درون مجموعه اعمال می‌کند.

طبق تعریف $\mathcal{F}$ مطمئن هستیم که رد پیشوندی‌ای که با محیط $\underline{\rho}$ شروع شود در مجموعه‌ی معنای S وجود دارد، چونکه به ازای هر محیط $\dot{\rho}$(نقطه به این خاطر است که با ρ خاص موجود

در فرض اشتباه گرفته نشود) حالت $\langle at[\![\mathsf{S}]\!], \acute{\rho}\rangle$ در هر اعمال تابع $\mathcal{F}$ روی هر مجموعه‌ی دلخواه وجود دارد. وقتی معنای S را به عنوان کوچک‌ترین نقطه ثابت $\mathcal{F}$ در نظر گرفته‌ایم پس مطمئن هستیم که آن مجموعه‌ای که کوچکترین نقطه ثابت است شامل رد پیشوندی $\langle at[\![\mathsf{S}]\!], \rho\rangle$ است.
این رد پیشوندی با اجرای $\mathcal{F}$ تحت تاثیر قرار می‌گیرد. اگر معنای B در یکی از اعمال های $\mathcal{F}$ غلط باشد، رد پیشوندی $\langle at[\![\mathsf{S}]\!], \rho\rangle\pi\langle aft[\![\mathsf{S}]\!], \rho'\rangle$ در معنای برنامه قرار خواهد گرفت و می‌توانیم بگوییم اجرای دستور با این محیط اولیه توقف پذیر است. می‌دانیم که طبق تعریف تابع به انتهای این رد پیشوندی چیزی اضافه نمی‌شود. از طرف دیگر هم با این محیط اولیه، با توجه به تعریف رد پیشوندی دیگری وجود ندارد که طولانی‌تر از رد پیشوندی مورد اشاره باشد.

در حالت دیگر اگر فرض کنیم هیچ گاه به حالتی نمی‌رسیم که در آن معنای B غلط باشد هم با فرض مسئله به تناقض می‌خوریم، چون در آن صورت تابع $\mathcal{F}$ مدام به طول دنباله‌هایی که با محیط ρ شروع می‌شوند می‌افزاید و این یک دنباله‌ی نامتناهی را خواهد ساخت. در صورتی که معنای B هیچ گاه صحیح نباشد، حداقل حالت $\langle at[\![\mathsf{S_t}]\!], \rho''\rangle$ به ته دنباله‌های پیشین اضافه خواهد شد و از این جهت مطمئن هستیم که دنباله‌ی نامتناهی گفته شده در معنای دستور حضور خواهد داشت.

پس با این تفاصیل، این مورد هم ثابت می‌شود.

$\blacktriangleright\blacktriangleright$ $\mathsf{S} =$ break; :

در تعریف تابع aft روی برچسب‌ها در [۶] این تعریف برای این دستور مشخص نیست! در [۷] که در مورد برچسب‌ها بحث شده، نویسنده‌ی [۶] گفته که در مورد آن بخش از تعاریف توابع مربوط به برچسب‌ها که تعریف نشده‌اند برداشت آزاد است و ما در اینجا سعی داریم معقول ترین برداشتی که نسبت به درکمان از این کار می‌توانیم داشته باشیم را بیان کنیم. مهم‌ترین چیزی که در مورد برچسب‌ها در مورد این دستور قرار است برقرار باشد این است که اگر این دستور بخشی از $\mathsf{S_t}$ در حلقه‌ی زیر باشد

$$\mathsf{S'} = \text{ while (B) } \mathsf{S_t}$$

در این صورت $aft[\![\mathsf{S'}]\!] = brk - to[\![\mathsf{S_t}]\!]$ را طبق تعریف داریم. انتظار می‌رود که $aft[\![\text{break;}]\!] = aft[\![\mathsf{S'}]\!]$ باشد. اینکه دستورات برنامه پس از اجرای break; از خارج(یا به عبارت بهتر بعد از) حلقه‌ی $\mathsf{S'}$ پی گرفته شود انتظار معقولی است از سیستمی که در حال توصیف رد اجرای برنامه‌های کامپیوتری است. البته در نظر گرفته شود که فرض کرده‌ایم که $\mathsf{S'}$ داخلی ترین حلقه‌ای است که break; درون آن جای دارد.

از پس این فرض‌های ما $aft[\![\text{break;}]\!] = break - to[\![\mathsf{S_t}]\!]$ نتیجه می‌شود و طبق تعریف معنای دستورات break; رد پیشوندی زیر در معنای این دستور وجود دارد

$$\langle at[\![\text{break;}]\!], \rho\rangle\langle aft[\![\text{break;}]\!], \rho\rangle$$

که نشانه‌ی توقف است.

$\blacktriangleright\blacktriangleright$ $\mathsf{S} = \{\mathsf{Sl''}\}$:

در این صورت توقف پذیری $\mathsf{Sl''}$ را از فرض استقرای استقرایی که روی لیست دستورات زده بودیم داریم پس $\{\mathsf{Sl''}\}$ هم توقف پذیر است.

در اینجا اثبات این طرف قضیه به پایان می‌رسد.

($\Leftarrow$) دوباره باید روی ساختار برنامه‌ها استقرا بزنیم و دوباره چون هر برنامه مساوی با یک لیست از دستورات است استقرا را ابتدا روی ساختار لیست دستورات و در دل آن روی ساختار دستورات استقرا می‌زنیم.

در این اثبات به غیر از یک حالت ساختار دستور، که دستور حلقه است، هر آنچه در مورد اثبات طرف راست قضیه گفتیم، به ما حکم را بدون نیاز به فرض نشان می‌دهد. بنابراین فقط در مورد اثبات همین یک مورد بحث می‌کنیم.

$\blacktriangleright\blacktriangleright$ $\mathsf{S} =$ while (B) $\mathsf{S_t}$:

اگر فرض کنیم این دستور به ازای محیط ρ در حالت اول متوقف شده، در واقع فرض کرده‌ایم در معنای این دستور رد پیشوندی $\langle at[\![\mathsf{S}]\!], \rho\rangle\pi\langle aft[\![\mathsf{S}]\!], \rho'\rangle\pi'$ وجود دارد. باید ثابت کنیم به

ازای π' دلخواه اگر رد پیشوندی $\langle at[\![\mathsf{S}]\!], \rho\rangle\pi\langle aft[\![\mathsf{S}]\!], \rho'\rangle\pi'$ داخل $\mathcal{S}^*[\![\mathsf{S}]\!]$ وجود داشته باشد آنگاه $\pi' = \epsilon$ برقرار است.

اگر برچسب $aft[\![\mathsf{S}]\!]$ در یک حالت در رد پیشوندی‌ای که گفتیم حضور داشته باشد، یعنی در یک دور اجرای حلقه عبارت بولی معنی غلط می‌داده که حالتی شامل این برچسب به یک رد پیشوندی چسبانده شده و این رد پیشوندی ساخته شده. از طرفی دیگر هم می‌دانیم که وقتی عبارت بولی حاضر در ساختار حلقه غلط شده، دیگر به ردهای پیشوندی داخل معنای حلقه چیزی اضافه نمی‌شود. بنابراین سناریوای جز $\pi' = \epsilon$ باقی نمی‌ماند. $\square$

پس با توجه به آنچه گفتیم می‌توانیم با خیال راحت توقف پذیری یک برنامه با یک محیط اولیه را معادل متناهی بودن همه‌ی ردهای پیشوندی‌ای بدانیم که با محیط متناظر با آن محیط اولیه شروع شده‌اند. اگر در صورت ارائه شده از وارسی مدل عبارت منظم R را با عبارت منظم ε جایگزین کنیم داریم:

$$\mathsf{P}, \underline{\rho} \models \mathsf{R} \Leftrightarrow (\{\underline{\rho}\} \times \mathcal{S}^*[\![\mathsf{P}]\!]) \subseteq \mathsf{prefix}(\mathcal{S}^r[\![\varepsilon \bullet (? : T)^*]\!]) = \mathsf{prefix}(\mathcal{S}^r[\![(? : T)^*]\!])$$

طبق تعریف معنای عبارات منظم، هر رد پیشوندی متناهی‌ای داخل مجموعه‌ی سمت راستی رابطه‌ی زیرمجموعه بودن قرار می‌گیرد. این یعنی اگر الگوریتمی برای بررسی $\mathsf{P}, \underline{\rho} \models \mathsf{R}$ داشته باشیم، این الگوریتم می‌تواند تشخیص دهد آیا برنامه‌ی P با محیط اولیه‌ی $\underline{\rho}$ متوقف می‌شود یا خیر! این یعنی الگوریتمی برای مسئله‌ی توقف پذیری، مسئله‌ای که تصمیم ناپذیر است! بنابراین چنین الگوریتمی نباید وجود داشته باشد که یعنی پیاده سازی‌ای برای شیوه‌ای که در حال بیانش هستیم وجود ندارد! ادامه‌ی کار روی همین تعریف پیش می‌رود و دو صورت دیگر هم که قرار است ساختارمندتر باشند در نهایت با این صورت معادل‌اند، هرچند که پس از رسیدن به بیان دو صورت دیگر خواهیم دید که همین صحبت‌هایی که در مورد این صورت می‌کنیم در مورد صورت‌های دیگر هم بدون در نظر گرفتن معادل بودن این ۳ صورت برقرار است.

فصل ۴

وارسی مدل منظم

در این فصل قرار است به بیانی ساختارمندتر از روش وارسی مدل برسیم. اهمیت ساختارمند تر بودن در این است که بیانی که در فصل پیش داشتیم تا پیاده سازی فاصله‌ی بسیاری دارد، چون همان‌طور که پیش‌تر گفته شد مجموعه‌ها موجودات ساختنی‌ای نیستند و کار با آن‌ها حین نوشتن برنامه‌ای کامپیوتری که قرار است پیاده‌سازی روش مورد بحث ما باشد را سخت می‌کند. ساختاری که در این فصل به صورت روش وارسی مدل اضافه می‌شود، ساختار عبارات منظم است و دلیل اینکه نام این فصل و تعریف اصلی‌ای که در این فصل قرار است ارائه شود "وارسی مدل منظم" است هم همین است که ساختار عبارات منظم به تعریف اضافه شده. از این رو پیش از اینکه به بیان وارسی مدل منظم بپردازیم، نیاز داریم که ابتدا به بررسی و تعریف برخی خواص عبارات منظم بپردازیم که در ادامه برای بیان وارسی مدل مورد نیاز هستند.

۱.۴ در مورد عبارات منظم

در این بخش ابتدا مفهوم هم‌ارز بودن را برای عبارات منظم تعریف می‌کنیم، سپس به سراغ تعریف دو تابع dnf و fstnxt می‌رویم.

۱.۱.۴ هم‌ارزی عبارات منظم

خیلی ساده هم‌ارزی بین دو عبارت منظم را با برابر بودن معنای آن دو تعریف می‌کنیم.

تعریف ۱.۴. (هم‌ارزی عبارات منظم): دو عبارت منظم R_1 و R_2
را هم‌ارز می‌گوییم اگر و تنها اگر شرط زیر برقرار باشد:

$$\mathcal{S}^r[\![\mathsf{R}_1]\!] = \mathcal{S}^r[\![\mathsf{R}_2]\!]$$

این هم‌ارزی را با $\mathsf{R_1} \Bumpeq \mathsf{R_2}$ نمایش می‌دهیم.

قضیه ۲.۴. هم‌ارزی $\Bumpeq$ تعریف شده روی مجموعه‌ی عبارات منظم یک رابطه‌ی هم‌ارزی است.

اثبات. برای هر عبارت منظم R داریم:

$$\mathcal{S}^r[\![\mathsf{R}]\!] = \mathcal{S}^r[\![\mathsf{R}]\!] \Rightarrow \mathsf{R} \Bumpeq \mathsf{R}$$

پس این رابطه انعکاسی است.
اگر $\mathsf{R_1}, \mathsf{R_2} \in \mathbb{R}$ آنگاه داریم:

$$\mathcal{S}^r[\![\mathsf{R_1}]\!] = \mathcal{S}^r[\![\mathsf{R_2}]\!] \to \mathcal{S}^r[\![\mathsf{R_2}]\!] = \mathcal{S}^r[\![\mathsf{R_1}]\!] \Rightarrow \mathsf{R_1} \Bumpeq \mathsf{R_2} \to \mathsf{R_2} \Bumpeq \mathsf{R_1}$$

پس این رابطه تقارنی هم هست.
اگر $\mathsf{R_1}, \mathsf{R_2}, \mathsf{R_3} \in \mathbb{R}$ داریم:

$$\mathcal{S}^r[\![\mathsf{R_1}]\!] = \mathcal{S}^r[\![\mathsf{R_2}]\!] \wedge \mathcal{S}^r[\![\mathsf{R_2}]\!] = \mathcal{S}^r[\![\mathsf{R_3}]\!] \to \mathcal{S}^r[\![\mathsf{R_1}]\!] = \mathcal{S}^r[\![\mathsf{R_3}]\!]$$

$$\Rightarrow \mathsf{R_1} \Bumpeq \mathsf{R_2} \wedge \mathsf{R_2} \Bumpeq \mathsf{R_3} \to \mathsf{R_1} \Bumpeq \mathsf{R_3}$$

□

۲.۱.۴ فرم نرمال فصلی

یک دسته از عبارات منظم هستند که به آن‌ها می‌گوییم فرم نرمال فصلی. در صورتی از وارسی مدل که در این فصل ارائه شده، مفهوم فرم نرمال فصلی حضور دارد، بنابراین باید به بحث در مورد آن، پیش از رسیدن به صورت جدید، بپردازیم.

تعریف ۳.۴. (فرم نرمال فصلی): عبارت منظم $\mathsf{R} \in \mathbb{R}$ را یک فرم نرمال فصلی می‌گوییم اگر و تنها اگر با فرض اینکه عبارات منظم بدون انتخاب $\mathsf{R_1}, \mathsf{R_2}, ..., \mathsf{R_n} \in \mathbb{R}^{\dagger}$ وجود داشته باشند که $\mathsf{R} = \mathsf{R_1} + \mathsf{R_2} + ...\mathsf{R_n}$.

در تعریف بالا به = دقت شود که با $\Bumpeq$ که در ادامه مورد بحث ماست فرق می‌کند. به سبک رایج منظور از = همان تساوی نحوی است.

در ادامه می‌خواهیم یک تابع به اسم dnf تعریف کنیم که یک عبارت منظم R را می‌گیرد و عبارت منظم $\mathsf{R'}$ را تحویل می‌دهد که یک فرم نرمال فصلی است و $\mathsf{R} \Bumpeq \mathsf{R'}$ برقرار است. ابتدا این تابع را به صورت استقرایی روی ساختار عبارات منظم تعریف می‌کنیم، سپس خاصیتی که گفتیم را درمورد آن ثابت می‌کنیم. اینْ اثبات این حقیقت خواهد بود که هر عبارت منظم با یک فرم نرمال فصلی هم ارز است.

تعریف ۴.۴. (تابع dnf): تابع dnf روی عبارات منظم به شکل زیر تعریف می‌شود:

$$\blacktriangleleft\ \mathsf{dnf}(\varepsilon) = \varepsilon$$

$$\blacktriangleleft\ \mathsf{dnf}(L : B) = L : B$$

$$\blacktriangleleft\ \mathsf{dnf}(R_1R_2) = \Sigma_{i=1}^{n_1}\Sigma_{j=1}^{n_2}R_1^iR_2^j$$

$$\text{where } R_1^1 + R_1^2 + ... + R_1^{n_1} = \mathsf{dnf}(R_1) \text{ and } R_2^1 + R_2^2 + ... + R_2^{n_2} = \mathsf{dnf}(R_2)$$

$$\blacktriangleleft\ \mathsf{dnf}(R_1 + R_2) = \mathsf{dnf}(R_1) + \mathsf{dnf}(R_2)$$

$$\blacktriangleleft\ \mathsf{dnf}(R^*) = ((R_1)^*(R_2)^*...(R_n)^*)^*$$

$$\text{where } \mathsf{dnf}(R) = R^1 + R^2 + ... + R^n$$

$$\blacktriangleleft\ \mathsf{dnf}(R^+) = \mathsf{dnf}(RR^*)$$

$$\blacktriangleleft\ \mathsf{dnf}((R)) = (\mathsf{dnf}(R))$$

قضیه ۵.۴. اگر $R \in \mathbb{R}$ آنگاه $\mathsf{dnf}(R)$ یک ترکیب نرمال فصلی است.

اثبات. همان طور که گفتیم روی ساختار R استقرا می‌زنیم.

$$\blacktriangleright\ R = \varepsilon :$$

$$\mathsf{dnf}(\varepsilon) = \varepsilon$$

که ε یک فرم نرمال فصلی است.

$$\blacktriangleright\ R = L : B :$$

$$\mathsf{dnf}(L : B) = L : B$$

که $L : B$ هم یک فرم نرمال فصلی است.

$$\blacktriangleright\ R = R_1R_2 :$$

فرض استقرا این خواهد بود که $\mathsf{dnf}(R_1) = R_1^1 + R_1^2 + ... + R_1^n$ و $\mathsf{dnf}(R_2) = R_2^1 + R_2^2 + ... + R_2^n$ درحالیکه $\mathsf{dnf}(R_1)$ و $\mathsf{dnf}(R_2)$ ترکیب نرمال فصلی هستند، یعنی هر R_1^i و هر R_2^j عضو $\mathbb{R}^\dagger$ است. طبق تعریف خواهیم داشت:

$$\mathsf{dnf}(R_1R_2) = \Sigma_{i=1}^{n_1}\Sigma_{j=1}^{n_2}R_1^iR_2^j$$

که طرف راست عبارت بالا یک ترکیب نرمال فصلی است، چون هر $R_1^iR_2^j$ یک عضو از $\mathbb{R}^\dagger$ است.

$\blacktriangleright\ R = R_1 + R_2 :$

فرض استقرا این خواهد بود که $\mathsf{dnf}(R_1)$ و $\mathsf{dnf}(R_2)$ ترکیب فصلی نرمال هستند پس $\mathsf{dnf}(R_1 + R_2)$ هم که برابر با $\mathsf{dnf}(R_1) + \mathsf{dnf}(R_2)$ است، ترکیب فصلی نرمال خواهد بود.

$\blacktriangleright\ R = R_1^* :$

طبق فرض استقرا داریم که $\mathsf{dnf}(R_1)$ یک ترکیب نرمال فصلی است. همین طور طبق تعریف dnf داریم

$$\mathsf{dnf}(R_1^*) = ((R_1^1)^*(R_1^2)^*...(R_1^n)^*)$$

که

$$\mathsf{dnf}(R_1) = R_1^1 + R_1^2 + ... + R_1^n$$

که اینکه $((R_1^1)^*(R_1^2)^*...(R_1^n)^*)$ یک فرم نرمال فصلی است مشخص است چون می‌دانیم در هیچ کدام از این R_1^i ها عملگر $+$ وجود ندارد و عملگر * و عملگر چسباندن هم تغییری در این وضع ایجاد نمی‌کنند.

$\blacktriangleright\ R = R_1^+ :$

طبق چیزهایی که از قبل داریم:

$$\mathsf{dnf}(R_1^+) = \mathsf{dnf}(R_1 R_1^*)$$

$$\mathsf{dnf}(R_1^*) = ((R_1^1)^*(R_1^2)^*...(R_1^n)^*)$$

که گیریم $R' = \mathsf{dnf}(R_1^*)$ که عضو $\mathbb{R}^\dagger$ است. همین طور فرض می‌کنیم:

$$R_1 = R_1^1 + ... + R_1^n$$

پس با توجه به تعریف dnf برای عملگر چسباندن خواهیم داشت:

$$\mathsf{dnf}(R_1^+) = \Sigma_{i=1}^n R_1^i R'$$

$\blacktriangleright\ R = (R_1) :$

طبق تعریف داریم:

$$\mathsf{dnf}((R_1)) = (\mathsf{dnf}(R_1))$$

طبق فرض استقرا $\mathsf{dnf}(R_1)$ یک ترکیب نرمال فصلی است، بنابراین $(\mathsf{dnf}(R_1)) = R' \in \mathbb{R}^\dagger$ هم یک ترکیب فصلی نرمال خواهد بود.

$\square$

گزاره‌ی دیگری که برای اثبات مانده برقرار بودن $\mathsf{dnf(R)} \doteqdot \mathsf{R}$ است. برای اثبات آن باید ابتدا قضیه‌ی زیر را اثبات کنیم که اثبات آن را ارجاع می‌دهیم به [۱۳].

قضیه ۶.۴. برای هر دو عبارت منظم $\mathsf{R_1}, \mathsf{R_2} \in \mathbb{R}$ داریم:

$$(\mathsf{R_1} + \mathsf{R_2})^* \doteqdot (\mathsf{R_1^*}\mathsf{R_2^*})^*$$

به عنوان نتیجه از قضیه‌ی بالا می‌توانیم با استفاده از یک برهان ساده به کمک استقرا روی اعداد طبیعی، حکم بالا را به جای ۲ برای تعداد دلخواه متناهی‌ای از عبارات منظم اثبات کنیم. در ادامه در واقع از این حکم در اثبات استفاده شده.

قضیه ۷.۴. برای هر $\mathsf{R} \in \mathbb{R}$ داریم:

$$\mathsf{dnf(R)} \doteqdot \mathsf{R}$$

اثبات. طبعا این اثبات با استقرا روی ساختار R انجام می‌شود. توجه شود که در هر حالت از استقرا عبارات منظم $\mathsf{R_1}, \mathsf{R_2}$ در ساختار R حضور دارند، فرض گرفته‌ایم که $\mathsf{dnf(R_1)} = \mathsf{R_1^1} + \mathsf{R_1^2} + \ldots + \mathsf{R_1^n}$ و $\mathsf{dnf(R_2)} = \mathsf{R_1^2} + \mathsf{R_2^2} + \ldots + \mathsf{R_2^m}$.

$$\blacktriangleright\ \mathsf{R} = \varepsilon :$$

$$\mathsf{dnf}(\varepsilon) = \varepsilon \Rightarrow \mathcal{S}^r[\![\mathsf{dnf}(\varepsilon)]\!] = \mathcal{S}^r[\![\varepsilon]\!]$$

$$\blacktriangleright\ \mathsf{R} = \mathsf{L : B}\ :$$

$$\mathsf{dnf(L : B)} = \mathsf{L : B} \Rightarrow \mathcal{S}^r[\![\mathsf{dnf(L : B)}]\!] = \mathcal{S}^r[\![\mathsf{L : B}]\!]$$

$$\blacktriangleright\ \mathsf{R} = \mathsf{R_1R_2}\ :$$

برای اثبات این حالت باید دو عبارت زیر را ثابت کنیم:

$$\mathcal{S}^r[\![\mathsf{R_1R_2}]\!] \subseteq \mathcal{S}^r[\![\mathsf{dnf(R_1R_2)}]\!]$$

$$\mathcal{S}^r[\![\mathsf{R_1R_2}]\!] \supseteq \mathcal{S}^r[\![\mathsf{dnf(R_1R_2)}]\!]$$

$(\supseteq)$:

فرض می‌کنیم $\langle \underline{\rho}, \pi \rangle$ یک عضو دلخواه از $\mathcal{S}^r[\![\mathsf{dnf(R_1R_2)}]\!]$ باشد. چون $\mathsf{dnf(R_1R_2)} = \Sigma_{\mathsf{i=1}}^{\mathsf{n_1}}\Sigma_{\mathsf{j=1}}^{\mathsf{n_2}}\mathsf{R_1^iR_2^j}$ پس داریم:

$$\exists k_1, k_2 : \pi \in \mathcal{S}^r[\![\mathsf{R_1^{k_1}R_2^{k_2}}]\!]$$

$$\Rightarrow \exists \pi_1, \pi_2\ s.t.\ \pi = \pi_1\pi_2, \langle \underline{\rho}, \pi_1 \rangle \in \mathcal{S}^r[\![\mathsf{R_1^{k_1}}]\!], \langle \underline{\rho}, \pi_2 \rangle \in \mathcal{S}^r[\![\mathsf{R_2^{k_2}}]\!]$$

با این وجود داریم:

$$\mathcal{S}^r[\![\mathsf{R}_1^{\mathsf{k}_1}]\!] \subseteq \mathcal{S}^r[\![\mathsf{R}_1]\!], \mathcal{S}^r[\![\mathsf{R}_2^{\mathsf{k}_2}]\!] \subseteq \mathcal{S}^r[\![\mathsf{R}_2]\!]$$
$$\Rightarrow \langle \underline{\rho}, \pi_1\pi_2\rangle = \langle \underline{\rho}, \pi\rangle \in \mathcal{S}^r[\![\mathsf{R}_1\mathsf{R}_2]\!]$$

$:(\subseteq)$

$$\langle \underline{\rho}, \pi\rangle \in \mathcal{S}^r[\![\mathsf{R}_1\mathsf{R}_2]\!]$$
$$\Rightarrow \exists \pi_1, \pi_2 : \pi = \pi_1\pi_2 \ s.t. \ \langle \underline{\rho}, \pi_1\rangle \in \mathcal{S}^r[\![\mathsf{R}_1]\!], \langle \underline{\rho}, \pi_2\rangle \in \mathcal{S}^r[\![\mathsf{R}_2]\!]$$

طبق فرض استقرا داریم:

$$\mathcal{S}^r[\![\mathsf{R}_1]\!] = \mathcal{S}^r[\![\mathsf{dnf}(\mathsf{R}_1)]\!], \mathcal{S}^r[\![\mathsf{R}_2]\!] = \mathcal{S}^r[\![\mathsf{dnf}(\mathsf{R}_2)]\!],$$
$$\Rightarrow \exists k_1, k_2 : \langle \underline{\rho}, \pi_1\rangle \in \mathcal{S}^r[\![\mathsf{R}_1^{\mathsf{k}_1}]\!], \langle \underline{\rho}, \pi_2\rangle \in \mathcal{S}^r[\![\mathsf{R}_2^{\mathsf{k}_2}]\!]$$
$$\Rightarrow \langle \underline{\rho}, \pi\rangle \in \mathcal{S}^r[\![\mathsf{R}_1^{\mathsf{k}_1}\mathsf{R}_2^{\mathsf{k}_2}]\!] \subseteq \mathcal{S}^r[\![\mathsf{dnf}(\mathsf{R}_1\mathsf{R}_2)]\!]$$

$$\blacktriangleright\ \mathsf{R} = \mathsf{R}_1 + \mathsf{R}_2 :$$
$$\mathcal{S}^r[\![\mathsf{dnf}(\mathsf{R}_1 + \mathsf{R}_2)]\!] =$$
$$\mathcal{S}^r[\![\mathsf{dnf}(\mathsf{R}_1) + \mathsf{dnf}(\mathsf{R}_2)]\!] =$$
$$\mathcal{S}^r[\![\mathsf{dnf}(\mathsf{R}_1)]\!] \cup \mathcal{S}^r[\![\mathsf{dnf}(\mathsf{R}_2)]\!] =$$

(به کمک فرض استقرا)

$$\mathcal{S}^r[\![\mathsf{R}_1]\!] \cup \mathcal{S}^r[\![\mathsf{R}_2]\!] =$$
$$\mathcal{S}^r[\![\mathsf{R}_1 + \mathsf{R}_2]\!]$$

$$\blacktriangleright\ \mathsf{R} = \mathsf{R}_1^* :$$
$$\mathcal{S}^r[\![\mathsf{dfn}(\mathsf{R}_1^*)]\!] =$$
$$\mathcal{S}^r[\![((\mathsf{R}_1^1)^*(\mathsf{R}_1^2)^*...(\mathsf{R}_1^{\mathsf{n}}))^*]\!] =$$

(طبق نتیجه‌ای که از قضیه‌ی قبل گرفتیم)

$$\mathcal{S}^r[\![(\mathsf{R}_1^1 + \mathsf{R}_1^2 + ... + \mathsf{R}_1^{\mathsf{n}})^*]\!] =$$
$$\mathcal{S}^r[\![(\mathsf{R}_1)^*]\!] =$$
$$\mathcal{S}^r[\![\mathsf{R}_1^*]\!]$$

$$\blacktriangleright\ \mathsf{R} = \mathsf{R}_1^+ :$$
$$\mathcal{S}^r[\![\mathsf{dfn}(\mathsf{R}_1^+)]\!] =$$
$$\mathcal{S}^r[\![\mathsf{dfn}(\mathsf{R}_1\mathsf{R}_1^*)]\!]$$

در اینجا عملگر چسباندن را داریم. در موردهای قبلی این را نشان دادیم که چه‌طور در این حالت حکم برقرار می‌شود. می‌توانیم همان اثبات را درمورد همین عبارت هم ببینیم و بگوییم:

$$\mathcal{S}^r[\![\mathsf{dfn}(\mathsf{R}_1\mathsf{R}_1^*)]\!] = \mathcal{S}^r[\![\mathsf{R}_1\mathsf{R}_1^*]\!] = \mathcal{S}^r[\![\mathsf{R}_1^+]\!]$$

$$\blacktriangleright\ \mathsf{R} = (\mathsf{R}_1) :$$
$$\mathcal{S}^r[\![\mathsf{dnf}((\mathsf{R}_1))]\!] =$$
$$\mathcal{S}^r[\![\mathsf{dnf}(\mathsf{R}_1)]\!] =$$

(طبق فرض استقرا:)

$$\mathcal{S}^r[\![\mathsf{R}_1]\!] =$$
$$\mathcal{S}^r[\![(\mathsf{R}_1)]\!]$$

□

۳.۱.۴ سر و دم عبارات منظم

در این بخش تعریف تابعی را روی عبارات منظم ارائه می‌کنیم که یک عبارت منظم را می‌گیرد و یک زوج از عبارات منظم را تحویل می‌دهد، سپس به بیان یک قضیه در مورد این تابع می‌پردازیم. این تابع را با fstnxt نشان می‌دهیم. قرار است این تابع یک عبارت منظم را بگیرد و آن را به این شکل تجزیه کند که اولین زوج موجود در عبارت منظم که انگار سر عبارت منظم است، از باقی آن که دم آن عبارت منظم می‌شود، جدا شود. تابع روی عبارات منظم تهی و عبارات منظمی که عملگر + را دارند تعریف نشده.

تعریف ۸.۴. (تابع سر و دم): تابع سر و دم را از نوع $\mathsf{fstnxt} : (\mathbb{R}^+ \cap \mathbb{R}^\dagger) \to \mathbb{R} \times \mathbb{R}$ به شکل زیر تعریف می‌کنیم:

$$\blacktriangleleft\ \mathsf{fstnxt}(\mathsf{L} : \mathsf{B}) = \langle \mathsf{L} : \mathsf{B}, \varepsilon \rangle$$
$$\blacktriangleleft\ \mathsf{fstnxt}(\mathsf{R}_1\mathsf{R}_2) = \mathsf{fstnxt}(\mathsf{R}_2)$$
$$\text{where } \mathsf{R}_1 \in \mathbb{R}_\varepsilon$$

$$\blacktriangleleft\ \mathsf{fstnxt}(\mathsf{R}_1\mathsf{R}_2) = (\!|\mathsf{R}_1^{\mathsf{n}} \in \mathbb{R}_\varepsilon\ ?\ \langle \mathsf{R}_1^{\mathsf{f}}, \mathsf{R}_2\rangle : \langle \mathsf{R}_1^{\mathsf{f}}, \mathsf{R}_1^{\mathsf{n}} \bullet \mathsf{R}_2\rangle|\!)$$

$$\text{where } \mathsf{R}_1 \notin \mathbb{R}_\varepsilon \text{and } \mathsf{fstnxt}(\mathsf{R}_1) = \langle \mathsf{R}_1^{\mathsf{f}}, \mathsf{R}_1^{\mathsf{n}}\rangle$$

$$\blacktriangleleft\ \mathsf{fstnxt}(\mathsf{R}^+) = (\!|\mathsf{R}^{\mathsf{n}} \in \mathbb{R}_\varepsilon\ ?\ \langle \mathsf{R}^{\mathsf{f}}, \mathsf{R}^*\rangle : \langle \mathsf{R}^{\mathsf{f}}, \mathsf{R}^{\mathsf{n}} \bullet \mathsf{R}^*\rangle|\!)$$

$$\text{where } \mathsf{fstnxt}(\mathsf{R}) = \langle \mathsf{R}^{\mathsf{f}}, \mathsf{R}^{\mathsf{n}}\rangle$$

$$\blacktriangleleft\ \mathsf{fstnxt}((\mathsf{R})) = \mathsf{fstnxt}(\mathsf{R})$$

از این تعریف قرار است در صورتی از وارسی مدل که در این فصل ارائه شده استفاده شود. یک قضیه در آخر این بخش آمده که مهم‌ترین نتیجه در مورد تابع سر و دم است. برای اثبات آن قضیه ابتدا یک گرامر برای $\mathbb{R}^+ \cap \mathbb{R}^\dagger$ می‌آوریم.

قضیه ۹.۴. گرامر زیر زبان $\mathbb{R}^+ \cap \mathbb{R}^\dagger$ را توصیف می‌کند.

$$\mathsf{R} \quad \in \quad \mathbb{R}^+ \cap \mathbb{R}^\dagger$$

$$\mathsf{R} \quad ::= \quad \mathsf{L} : \mathsf{B} \mid \varepsilon\mathsf{R}_2 \mid \mathsf{R}_1\varepsilon \mid \mathsf{R}_1\mathsf{R}_2 \mid \mathsf{R}_1^+ | (\mathsf{R}_1)$$

اثبات. نام مجموعه‌ی عبارات منظم تولید شده با گرامر بالا را $\mathbb{R}'$ می‌گذاریم. باید ثابت کنیم $\mathbb{R}' = \mathbb{R}^+ \cap \mathbb{R}^\dagger$ که برای این باید ثابت کنیم این دو مجموعه زیر مجموعه‌ی یکدیگر هستند. برای اثبات $\mathbb{R}' \subseteq \mathbb{R}^+ \cap \mathbb{R}^\dagger$ می‌توانیم روی ساختار گرامر بالا استقرا بزنیم:

$$\blacktriangleright\ \mathsf{R} = \mathsf{L} : \mathsf{B}\ :$$

به وضوح $\mathsf{L} : \mathsf{B} \in \mathbb{R}^+ \cap \mathbb{R}^\dagger$ است و این را از گرامر $\mathbb{R}^+$ و $\mathbb{R}^\dagger$ می‌توانیم ببینیم.

$$\blacktriangleright\ \mathsf{R} = \varepsilon\mathsf{R}_2\ :$$

با فرض اینکه $\mathsf{R}_2 \in \mathbb{R}^+ \cap \mathbb{R}^\dagger$ که فرض استقراست، طبق گرامر $\mathbb{R}^+$ داریم $\varepsilon\mathsf{R}_2 \in \mathbb{R}^+$ و طبق گرامر $\mathbb{R}^\dagger$ داریم چون $\mathsf{R}_2 \in \mathbb{R}^\dagger$ و $\varepsilon \in \mathbb{R}^\dagger$ پس $\varepsilon\mathsf{R}_2 \in \mathbb{R}^\dagger$. پس داریم $\varepsilon\mathsf{R}_2 \in \mathbb{R}^+ \cap \mathbb{R}^\dagger$.

$$\blacktriangleright\ \mathsf{R} = \mathsf{R}_1\varepsilon\ :$$

مشابه مورد قبل ثابت می‌شود.

$$\blacktriangleright\ \mathsf{R} = \mathsf{R}_1\mathsf{R}_2\ :$$

طبق فرض استقرا داریم $\mathsf{R}_1, \mathsf{R}_2 \in \mathbb{R}^\dagger \cap \mathbb{R}^+$ و در هر دو گرامر عملگر چسباندن را داریم. پس این مورد هم اثبات می‌شود.

$$\blacktriangleright\ \mathsf{R} = \mathsf{R}_1^+\ :$$

مثل مورد قبل چون عملگر $^+$ در هر دو گرامر هست مثل مورد قبل به کمک فرض استقرا اثبات می‌شود.

$$\blacktriangleright \ \mathsf{R} = (\mathsf{R}_1) \ :$$

مثل مورد قبل اثبات می‌شود.

در اینجا اثبات $\mathbb{R}' \subseteq \mathbb{R}^+ \cap \mathbb{R}^\dagger$ کامل می‌شود. حال به سراغ اثبات $\mathbb{R}^+ \cap \mathbb{R}^\dagger \subseteq \mathbb{R}'$ می‌رویم.

برای اثبات این بخش با فرض اینکه $\mathsf{R} \in \mathbb{R}^+ \cap \mathbb{R}^\dagger$ روی ساختار اعضای $\mathbb{R}^\dagger$ استقرا می‌زنیم(این اثبات می‌توانست با استقرا روی ساختار اعضای $\mathbb{R}^+$ هم انجام شود).

$$\blacktriangleright \ \mathsf{R} = \varepsilon \ :$$

چون $\varepsilon \notin \mathbb{R}^+ \cap \mathbb{R}^\dagger$ پس اصلا این مورد باطل است و در مورد آن نیازی به ارائه‌ی اثبات نیست.

$$\blacktriangleright \ \mathsf{R} = \mathsf{L} : \mathsf{B} \ :$$

در این صورت طبق گرامر $\mathbb{R}'$ داریم $\mathsf{R} \in \mathbb{R}'$.

$$\blacktriangleright \ \mathsf{R} = \mathsf{R}_1\mathsf{R}_2 \ :$$

اینجا هم با توجه به اینکه طبق فرض استقرا $\mathsf{R}_1, \mathsf{R}_2 \in \mathbb{R}'$ مثل مورد قبل چون $+$ در گرامر $\mathbb{R}'$ حضور دارد، حکم ثابت می‌شود.

$$\blacktriangleright \ \mathsf{R} = \mathsf{R}_1^* \ :$$

چون به ازای هیچ عبارت منظم R_1ای ${\mathsf{R}_1}^*$ داخل $\mathbb{R}^+$ نمی‌افتد پس بررسی این مورد هم مورد نیاز نیست.

$$\blacktriangleright \ \mathsf{R} = \mathsf{R}_1^+ \ :$$

مثل عملگر $+$ با توجه به فرض استقرا و اینکه $^+$ در گرامر $\mathbb{R}'$ حضور دارد، این مورد هم اثبات می‌شود.

$$\blacktriangleright \ \mathsf{R} = (\mathsf{R}_1) \ :$$

مثل مورد قبلی است.

□

ساختاری که تابع سر و دم روی آن تعریف شده با این ساختار ریخت متفاوتی دارد و البته لزومی هم ندارد که یکی باشند. ساختاری که در قضیه‌ی قبل ارائه کرده‌ایم در [۶] نیامده و خودمان با هدف اثبات قضیه‌ی بعدی، آن را در اینجا ارائه کرده‌ایم.

قضیه ۱۰.۴. برای هر عبارت منظم $R \in \mathbb{R}^{+} \cap \mathbb{R}^{\dagger}$ اگر $fstnxt(R) = \langle L : B, R' \rangle$ آنگاه $R' \in \mathbb{R}^{\dagger}$ و $R \doteqdot L : B \bullet R'$.

اثبات. اثبات را باید با استقرا روی ساختار عبارات منظم عضو $\mathbb{R}^{+} \cap \mathbb{R}^{\dagger}$ زد.

$$\blacktriangleright\ R = L : B;$$

در این حالت طبق تعریف تابع سر و دم داریم $fstnxt(R) = \langle L : B, \varepsilon \rangle$. از طرف دیگر $\mathcal{S}^r[\![L : B \bullet \varepsilon]\!] = \mathcal{S}^r[\![L : B]\!]$ را داریم و ε عضو $\mathbb{R}^{\dagger}$ است.

$$\blacktriangleright\ R = \varepsilon R_2;$$

طبق تعریف تابع سر و دم داریم $fstnxt(\varepsilon R_2) = fstnxt(R_2)$. فرض استقرا این است که اگر $fstnxt(R_2) = \langle L : B, R_2' \rangle$ آنگاه $R_2' \in \mathbb{R}^{\dagger}$ و $R_2 \doteqdot L : B \bullet R_2'$. پس $fstnxt(R) = \langle L : B, R_2' \rangle$ که همان طور که گفتم طبق فرض استقرا $R_2' \in \mathbb{R}^{\dagger}$ و از طرف دیگر:

$$R = \varepsilon R_2 \doteqdot R_2 \doteqdot L : B \bullet R_2'$$

$$\blacktriangleright\ R = R_1 \varepsilon;$$

در این حالت امکان ندارد $R_1 = \varepsilon$ باشد، چون در آن صورت خواهیم داشت $R = \varepsilon\varepsilon \in \mathbb{R}_{\varepsilon}$ که تناقض است چون $\varepsilon\varepsilon$ در دامنه‌ی تابع سر و دم نیست. طبق تعریف سر و دم اگر داشته باشیم $fstnxt(R_1) = \langle L : B, R_1' \rangle$
در آن صورت بنا بر این که $R_1' \in \mathbb{R}_{\varepsilon}$ برقرار هست یا نه دو حالت را داریم:

$$\blacktriangleright\blacktriangleright\ R_1' \in \mathbb{R}_{\varepsilon} :$$

در این صورت $fstnxt(R) = \langle L : B, \varepsilon \rangle$ برقرار است. چون $R \in (\mathbb{R}^{+} \cap \mathbb{R}^{\dagger})$ پس چون R_2 زیر رشته‌ی R است، پس $R_2 \in \mathbb{R}^{\dagger}$ برقرار است. اصلا در این صورت $R = L : B\ \varepsilon$ برقرار خواهد بود. بنابراین $L : B\ \varepsilon \doteqdot L : B \bullet \varepsilon$ هم بدیهی خواهد بود.

$$\blacktriangleright\blacktriangleright\ R_1' \notin \mathbb{R}_{\varepsilon} :$$

در این صورت $fstnxt(R) = \langle L : B, R_1' \bullet \varepsilon \rangle$ طبق تعریف سر و دم برقرار است. چون $R \in \mathbb{R}^{+} \cap \mathbb{R}^{\dagger}$، پس زیر رشته‌های آن نیز عملگر $+$ را نخواهند داشت، پس $R \in \mathbb{R}^{\dagger}$. در اینجا نیز واضح است که:

$$L : B \bullet R_1' \bullet \varepsilon \doteqdot R_1 \varepsilon = R$$

$$\blacktriangleright\ R = R_1 R_2;$$

اگر یکی از R_1 و R_2 برابر ε باشد که حالات بالا را داریم و اگر هر دو برابر ε باشند هم که اصلا به تناقض می‌خوریم چون در این صورت دیگر در $\mathbb{R}^+$ این عبارت منظم را نداریم. پس تنها یک حالت می‌ماند و آن اینکه هیچ یک از این دو عبارت منظم تهی نباشند. باز هم اگر فرض کنیم $\mathsf{fstnxt}(R_1) = \langle L : B, R_1' \rangle$ مسئله بنا به اینکه $R_1' \in \mathbb{R}_\varepsilon$ برقرار هست یا خیر افراز می‌شود، مانند حالت قبل.

$$\blacktriangleright\blacktriangleright\ R_1' \notin \mathbb{R}_\varepsilon :$$

در این صورت $\mathsf{fstnxt}(R) = \langle L : B, R_1' \bullet R_2 \rangle$. مانند آنچه بالاتر استدلال کردیم خواهیم داشت $R_1' \bullet R_2 \in \mathbb{R}^\dagger$. علاوه‌بر این طبق فرض استقرا داریم $L : B \bullet R_1' \doteqdot R_1$، پس:

$$L : B \bullet R_1' \bullet R_2 \doteqdot R_1 \bullet R_2 = R$$

(عملگر چسباندن شرکت پذیر است). پس این حالت اثبات می‌شود.

$$\blacktriangleright\blacktriangleright\ R_1' \in \mathbb{R}_\varepsilon :$$

در این صورت $\mathsf{fstnxt}(R) = \langle L : B, R_2 \rangle$. مثل حالت‌های قبل ثابت می‌شود که $R_2 \in \mathbb{R}^\dagger$ و اینکه داریم:

$$R = L : B \bullet R_2 \Rightarrow L : B \bullet R_2 \doteqdot R$$

$$\blacktriangleright\ R = R_1^+;$$

با فرض اینکه $\mathsf{fstnxt}(R_1) = \langle L : B, R_1' \rangle$ بنا به اینکه $R_1' \in \mathbb{R}_\varepsilon$ برقرار باشد یا نه، دو حالت خواهد بود:

$$\blacktriangleright\blacktriangleright\ R_1' \in \mathbb{R}_\varepsilon :$$

در این صورت طبق تعریف تابع سر و دم $\mathsf{fstnxt}(R) = \langle L : B, R_1^* \rangle$ را داریم. R_1^* عضو $\mathbb{R}^\dagger$ خواهد بود چونکه داخل R_1' و $L : B$ عملگر $+$ وجود ندارد و جای دیگری از این عبارت منظم وجود ندارد که در آن بتوان وجود این عملگر را متصور شد. همین طور داریم:

$$\mathsf{fstnxt}(R_1) = \langle L : B, R_1' \rangle \rightarrow R_1' \in \mathbb{R}^\dagger \ \wedge\ L : B \bullet R_1' \doteqdot R_1$$

(عبارت بالا فرض استقراست.)

$$R_1^* \doteqdot (L : B \bullet R_1')^* \doteqdot (L : B \bullet \varepsilon)^* \doteqdot (L : B)^*$$

(هم‌ارزی وسطی به خاطر این است که R_1' عضو $\mathbb{R}^\dagger$ است. اگر یکی از دو هم‌ارزی دیگر هم برقرار نباشند کلا عملگر $*$ خوش تعریف نخواهد بود، پس این دو هم‌ارزی باید برقرار باشند.)

$$\Rightarrow L : B \bullet R_1^* \doteqdot L : B \bullet (L : B)^* \doteqdot (L : B)^+$$

$$\blacktriangleright\blacktriangleright\ \mathsf{R}'_1 \notin \mathbb{R}_\varepsilon\ :$$

با توجه به تعریف تابع سر و دم و فرض استقرا که چند خط بالاتر بیان شده، در این حالت داریم $\mathsf{fstnxt}(\mathsf{R}) = \langle \mathsf{L} : \mathsf{B}, \mathsf{R}'_1 \bullet \mathsf{R}^*_1 \rangle$. به همان دلیل مورد قبلی می‌دانیم که R^*_1 عضو $\mathbb{R}^\dagger$ است و طبق فرض استقرا داریم $\mathsf{R}'_1 \in \mathbb{R}^\dagger$. بنابراین داریم $\mathsf{R}'_1 \bullet \mathsf{R}^*_1 \in \mathbb{R}^\dagger$.

باز هم با استفاده از فرض استقرا داریم:

$$\mathsf{L} : \mathsf{B} \bullet \mathsf{R}'_1 \doteqdot \mathsf{R}_1$$

$$\Rightarrow \mathsf{L} : \mathsf{B} \bullet \mathsf{R}'_1 \bullet \mathsf{R}^*_1 \doteqdot \mathsf{R}_1\mathsf{R}^*_1 \doteqdot \mathsf{R}^+_1$$

$$\blacktriangleright\ \mathsf{R} = (\mathsf{R}_1)\ :$$

از فرض استقرا نتیجه می‌شود. □

این بخش در این قسمت به پایان می‌رسد. الان ابزارهای کافی برای بیان روش وارسی مدل به شکل جدیدی که مد نظر است را داریم.

۲.۴ وارسی مدل منظم

همان‌طور که گفتیم، در این فصل می‌خواهیم یک صورت معادل با صورتی که در فصل پیش برای روش وارسی مدل آورده شده بود را ارائه کنیم و تا به اینجای فصل صرفا به معرفی چند مفهوم که برای بیان این صورت جدید احتیاج داشتیم پرداخته‌ایم. در این یخش ابتدا این صورت جدید را بیان می‌کنیم و سپس اثبات می‌کنیم که صورت جدید با صورت قبلی معادل است. همان‌طور که پیش‌تر هم اشاره شد، تفاوت این بیان با بیان قبلی این است که این بیان روی ساختار عبارات منظم تعریف شده در حالیکه صورت قبلی ساختاری نداشت.

۱.۲.۴ صورت

در نهایت برای تعریف صورت به یک تابع $\mathcal{M}$ احتیاج داریم که در ورودی‌اش یک زوج متشکل از یک محیط اولیه و یک عبارت منظم را در کنار یک برنامه می‌گیرد و در خروجی همه‌ی ردهای پیشوندی موجود در معنای برنامه را که با عبارت منظم سازگار هستند، داخل یک مجموعه بر می‌گرداند. اما در این بین، مفهوم سازگاری یک رد پیشوندی با یک عبارت منظم چگونه مشخص می‌شود؟ این نکته‌ای است که تا به حال در مورد آن بحث نکرده‌ایم و الان می‌خواهیم تعریف تابع $\mathcal{M}^t$ را با این هدف به بحث وارد کنیم. البته این تابع قرار است یک ویژگی بیشتر هم داشته باشد و

این ویژگی این است که اگر عبارت منظم با رد پیشوندی سازگار نباشد، به ما می‌گوید که از کجای عبارت منظم ناسازگاری وجود داشته است و اگر هم این دو با هم سازگار باشند، این تابع به ما نشان می‌دهد که عبارت منظم تا کجا بررسی شده(فهمیدن این موضوع با نگاه به تعریف ساده‌تر است و البته نباید فراموش کرد که سازگاری‌ای که داریم، قرار است پیرو صورت قبلی باشد که در آن اگر طول یک عبارت منظم از یک رد پیشوندی بیشتر باشد و صرفا تا اتمام طول رد پیشوندی سازگاری بین این دو برقرار باشد، در نهایت هم سازگار حسابشان می‌کنیم، به عبارت دیگر داریم از نقش عملگر prefix در صورت قبلی صحبت می‌کنیم).

تعریف ۱۱.۴ . (وارسی‌گر رد پیشوندی): به تابع $\mathcal{M}^t$ از نوع $(\underline{\mathbb{EV}} \times \mathbb{R}^{\dagger}) \to \mathfrak{S}^{+\infty} \to (\mathbb{B} \times \mathbb{R}^{\dagger})$ وارسی‌گر رد پیشوندی می‌گوییم. این تابع ضابطه‌ی زیر را دارد:

$$\blacktriangleleft\ \mathcal{M}^t\langle \underline{\rho}, \varepsilon\rangle \pi = \langle T, \varepsilon\rangle$$

(برای هر عضو دیگر $\mathbb{R}_\varepsilon$ هم ضابطه‌ی بالایی برقرار است. دو ضابطه‌ی پایینی برای عبارات منظم عضو $\mathbb{R}^{+} \cap \mathbb{R}^{\dagger}$ هستند.)

$$\blacktriangleleft\ \mathcal{M}^t\langle \underline{\rho}, \mathsf{R}\rangle \epsilon = \langle T, \mathsf{R}\rangle$$

$$\blacktriangleleft\ \mathcal{M}^t\langle \underline{\rho}, \mathsf{R}\rangle \pi = (\!|\langle \underline{\rho}, \langle l_1, \rho_1\rangle\rangle \in \mathcal{S}^r[\![\mathsf{L} : \mathsf{B}]\!]\ ?\ \mathcal{M}^t\langle \underline{\rho}, \mathsf{R}'\rangle \pi'\ :\ \langle F, \mathsf{R}\rangle|\!)$$
$$\text{while } \pi = \langle l_1, \rho_1\rangle \pi' \text{ and } \langle \mathsf{L} : \mathsf{B}, \mathsf{R}'\rangle = \mathsf{fstnxt}(\mathsf{R})$$

برای اینکه به $\mathcal{M}$ برسیم به معرفی یک تابع دیگر هم می‌پردازیم. این تابع را با $\mathcal{M}^{\dagger}$ نشان می‌دهیم و در واقع همان کاری را که $\mathcal{M}$ قرار است به ازای همه‌ی عبارات منظم انجام دهد، این تابع روی عبارات منظمی که $+$ ندارند انجام می‌دهد.

تعریف ۱۲.۴. (وارسی مدل منظم محدود به $\mathbb{R}^{\dagger}$): به تابع $\mathcal{M}^{\dagger}$ از نوع $(\underline{\mathbb{EV}} \times \mathbb{R}^{\dagger}) \to P(\mathfrak{S}^{+\infty}) \to P(\mathfrak{S}^{+\infty} \times \mathbb{R}^{\dagger})$ می‌گوییم وارسی مدل منظم محدود به $\mathbb{R}^{\dagger}$. ضابطه‌ی این تابع به شکل زیر است:

$$\mathcal{M}^{\dagger}\langle \underline{\rho}, \mathsf{R}\rangle \Pi = \{\langle \pi, \mathsf{R}'\rangle | \pi \in \Pi \wedge \mathcal{M}^t\langle \underline{\rho}, \mathsf{R}\rangle \pi = \langle T, \mathsf{R}'\rangle\}$$

حالا خود $\mathcal{M}$ را تعریف می‌کنیم. تعریف این تابع چیزی نیست جز اجتماع گرفتن از خروجی تابع بالا به ازای عبارات منظمی که در فرم نرمال عبارت منظم ورودی تابع حضور دارند. البته اطلاعاتمان از عبارات منظم در هر زوجی که در خروجی $\mathcal{M}^{\dagger}$ وجود دارد حذف می‌شود، یعنی صرفا ردهای پیشوندی را در مجموعه‌ای که خروجی $\mathcal{M}$ است، داریم.

تعریف ۱۳.۴. (وارسی مدل منظم):
تابع $\mathcal{M}$ را از نوع $(\underline{\mathbb{EV}} \times \mathbb{R}) \rightarrow P(\mathfrak{S}^{+\infty}) \rightarrow (\underline{\mathbb{EV}} \times P(\mathfrak{S}^{+\infty}))$ وارسی مدل منظم می‌گوییم که ضابطه‌ی زیر را دارد:

$$\mathcal{M}\langle\underline{\rho}, \mathsf{R}\rangle\Pi = \bigcup_{i=1}^{n}\{\langle\underline{\rho}, \pi\rangle | \exists \mathsf{R}' \in \mathbb{R}:\ \langle\pi, \mathsf{R}'\rangle \in \mathcal{M}^{\dagger}\langle\underline{\rho}, \mathsf{R_i}\rangle\Pi\}$$

$$\text{while } \mathsf{dnf}(\mathsf{R}) = \mathsf{R_1} + \mathsf{R_2} + \ldots + \mathsf{R_n}$$

در این صورت اگر خاصیت $\mathsf{R} \in \mathbb{R}$ در محیط اولیه‌ی $\underline{\rho}$ برای برنامه‌ی P برقرار باشد، می نویسیم

$$\mathsf{P}, \underline{\rho} \models_r \mathsf{R}$$

و برقرار بودن این رابطه با شرط زیر تعریف می‌شود:

$$\mathsf{P}, \underline{\rho} \models_r \mathsf{R} \iff \{\underline{\rho}\} \times \mathcal{S}^*[\![\mathsf{P}]\!] \subseteq \mathcal{M}\langle\underline{\rho}, \mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!]$$

با این تعریف در واقع زمانی می‌توانیم بگوییم، برنامه‌ی P خاصیت R دارد که $\mathcal{M}\langle\underline{\rho}, \mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!] = \{\underline{\rho}\} \times \mathcal{S}^*[\![\mathsf{P}]\!]$. در واقع انگار تابع $\mathcal{M}\langle\underline{\rho}, \mathsf{R}\rangle$ مثل یک صافی روی مجموعه‌ی $\mathcal{S}^*[\![\mathsf{P}]\!]$ عمل می‌کند، پس خروجی این مجموعه زیر مجموعه‌ی $\{\underline{\rho}\} \times \mathcal{S}^*[\![\mathsf{P}]\!]$ است.

قضیه ۱۴.۴. برای هر برنامه‌ی P، محیط اولیه‌ی $\underline{\rho}$ و عبارت منظم R داریم:

$$\mathcal{M}\langle\underline{\rho}, \mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!] \subseteq \{\underline{\rho}\} \times \mathcal{S}^*[\![\mathsf{P}]\!]$$

اثبات. اگر زوج $\langle\underline{\rho}, \pi\rangle$ عضو $\mathcal{M}\langle\underline{\rho}, \mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!]$ باشد، طبق تعریف $\mathcal{M}$، یعنی اگر $\mathsf{dnf}(\mathsf{R}) = \mathsf{R_1} + \mathsf{R_2} + \ldots + \mathsf{R_n}$ باشد و عبارت منظم $\mathsf{R}' \in \mathbb{R}$ و عدد i بین ۱ و n وجود دارند که:

$$\langle\pi, \mathsf{R}'\rangle \in \mathcal{M}^{\dagger}\langle\underline{\rho}, \mathsf{R_i}\rangle\mathcal{S}^*[\![\mathsf{P}]\!]$$

که طبق تعریف $\mathcal{M}^{\dagger}$ یعنی:

$$\pi \in \mathcal{S}^*[\![\mathsf{P}]\!] \wedge \mathcal{M}^t\langle\underline{\rho}, \mathsf{R_i}\rangle\pi = \langle T, \mathsf{R}'\rangle$$

که از $\pi \in \mathcal{S}^*[\![\mathsf{P}]\!]$ می‌توان نتیجه گرفت $\langle\underline{\rho}, \pi\rangle \in \{\underline{\rho}\} \times \mathcal{S}^*[\![\mathsf{P}]\!]$.

$\square$

مجموعه‌ی معنای یک برنامه را می‌توان به مجموعه‌ای از دسته‌ها افراز کرد، که در هر یک از این دسته‌ها ردهای پیشوندی‌ای حضور دارند که وضعیت اول آن‌ها یکسان است. قاعدتا در هر یک از این دسته‌ها باید وضعیت‌های بعدی هم، در صورت وجود، به طور موازی با یکدیگر یکسان

باشند، یعنی مثلا در یک مجموعه از افراز توصیف شده، همه‌ی ردهای پیشوندی‌ای که عضو دوم دارند، عضو دومشان با هم برابر است. این حرف برای عضو سوم و چهارم و غیره هم برقرار است. در هر دسته از این افراز یک رد پیشوندی ماکسیمال وجود خواهد داشت، که توصیف تمام و کمال برنامه، در اجرا با وضعیت اول مختص آن مجموعه است.

حال برای هر برنامه‌ی P که خاصیت R در مورد آن مورد بررسی است، می‌توانیم همین افراز را روی مجموعه‌ی $\mathcal{M}\langle\underline{\rho},\mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!]$ در نظر بگیریم. اگر در هر دسته از این افراز، رد پیشوندی ماکسیمال همین افراز روی $\mathcal{S}^*[\![\mathsf{P}]\!]$ (یعنی معنای برنامه) در دسته‌ی متناظر(یعنی دسته‌ای که از عضو ماکسیمالش روی خروجی $\mathcal{M}$ صحبت کردیم، منظور از متناظر این است که هر دو دسته در هر دو افراز ردهای پیشوندی با وضعیت اول یکسان دارند) وجود داشته باشد، در این صورت پس حتما $\mathcal{M}\langle\underline{\rho},\mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!] = \{\underline{\rho}\} \times \mathcal{S}^*[\![\mathsf{P}]\!]$. اگر دسته‌ای از افراز روی $\mathcal{M}\langle\underline{\rho},\mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!]$ دارای عضو ماکسیمال متفاوتی نسبت به همان دسته روی $\mathcal{S}^*[\![\mathsf{P}]\!]$ باشد، قطعا این عضو ماکسیمال کوتاه‌تر از عضو ماکسیمال همان دسته در $\mathcal{S}^*[\![\mathsf{P}]\!]$ است و این یعنی ناسازگاری‌ای با عبارت منظم مورد بررسی وجود داشته است. محل وقوع ناسازگاری را تابع $\mathcal{M}^\dagger$ می‌تواند به ما بگوید. محل ناسازگاری، عبارت منظمی است که زوج عضو ماکسیمال دسته‌ی مورد نظر در خروجی یکی از اعمال های $\mathcal{M}^\dagger$ روی بخش‌های مختلف $\mathsf{dnf}(\mathsf{R})$ است.

۲.۲.۴ درستی و تمامیت

حال به اثبات معادل بودن صورت جدید با صورت قبلی می‌پردازیم. در [۶] این اثبات که یک قضیه‌ی دوطرفه است، تحت دو قضیه به نام‌های درستی و تمامیت آمده. درستی به این معناست که اگر یک بررسی با روش جدید انجام شود، نتیجه‌ای یکسان با انجام بررسی برای همان برنامه و همان عبارت منظم در صورت قبلی دارد و تمامیت نیز عکس آن است، یعنی هر بررسی‌ای که با صورت قبلی انجام شده، نتیجه‌ی یکسانی با انجام همان بررسی در صورت جدید دارد.

نگارنده‌ی این پایان نامه، به درستی دو اثبات موجود در [۶] بسیار بد بین است! در اثبات تمامیت، برهان به شکل عجیبی بی‌ربط است و در اثبات قضیه درستی، ایرادات فنی ریزی در جزئیات وجود دارد که با تعاریف در تناقض است. از این رو برهان‌هایی که در اینجا آورده‌ایم جدید هستند.

قضیه ۱۵.۴. (قضیه درستی): اگر P یک برنامه، R یک عبارت منظم و $\underline{\rho}$ یک محیط اولیه باشند، آنگاه داریم:

$$\mathsf{P}, \underline{\rho} \models_r \mathsf{R} \Rightarrow \mathsf{P}, \underline{\rho} \models \mathsf{R}$$

اثبات. طبق تعریف دو صورت، باید با فرض اینکه داریم:

$$\{\underline{\rho}\} \times \mathcal{S}^*[\![\mathsf{P}]\!] \subseteq \mathcal{M}\langle\underline{\rho},\mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!]$$

ثابت کنیم:

$$\{\underline{\rho}\} \times \mathcal{S}^*[\![\mathsf{P}]\!] \subseteq \mathsf{prefix}(\mathcal{S}^{\mathsf{r}}[\![\mathsf{R} \bullet (?:T)^*]\!])$$

در این راستا، می‌توانیم گزاره‌ی زیر را ثابت کنیم که از گزاره‌ی قبلی قوی‌تر است و آن را نتیجه می‌دهد:

$$\mathcal{M}\langle\underline{\rho}, \mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!] \subseteq \mathsf{prefix}(\mathcal{S}^{\mathsf{r}}[\![\mathsf{R} \bullet (?:T)^*]\!])$$

در مورد عبارت منظم R فرض می‌کنیم $\mathsf{dnf}(\mathsf{R}) = \mathsf{R}_1 + \mathsf{R}_2 + \dots + \mathsf{R}_n$، که هر R_i ذکر شده، عضو $\mathbb{R}^{\dagger}$ است. فرض می‌کنیم $\langle\underline{\rho}, \pi\rangle \in \mathcal{M}\langle\underline{\rho}, \mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!]$ آنگاه وجود دارد k و R' که $\langle\pi, \mathsf{R}'\rangle \in \mathcal{M}^{\dagger}\langle\underline{\rho}, \mathsf{R}_k\rangle\mathcal{S}^*[\![\mathsf{P}]\!]$ که طبق تعریف $\mathcal{M}^{\dagger}$ خواهیم داشت:

$$\pi \in \mathcal{S}^*[\![\mathsf{P}]\!] \wedge \mathcal{M}^t\langle\underline{\rho}, \mathsf{R}_k\rangle\pi = \langle T, \mathsf{R}'\rangle$$

که طرف چپ گزاره‌ی عطفی بالا، در فرض بود و در ادامه‌ی کار با طرف راست این عبارت پیش می‌رویم. یعنی:

$$\mathcal{M}^t\langle\underline{\rho}, \mathsf{R}_k\rangle\pi = \langle T, \mathsf{R}'\rangle$$

در مورد R_k دو حالت داریم، یا $\mathsf{R}_k \simeq \varepsilon$ برقرار است، یا اینگونه نیست($\mathsf{R}_k \not\simeq \varepsilon$).

▶ $\mathsf{R}_k \simeq \varepsilon$:

در این صورت می‌توانیم ثابت کنیم

$$\mathsf{prefix}(\mathcal{S}^{\mathsf{r}}[\![\mathsf{R} \bullet (?:T)^*]\!]) = \{\underline{\rho}\} \times \mathbb{S}^+$$

با توجه به پخش پذیری عملگر چسباندن روی عملگر انتخاب، که پیش‌تر ثابت کردیم، داریم:

$$\mathsf{R} \bullet (?:T)^* \simeq (\mathsf{R}_1 + \mathsf{R}_2 + \dots + \mathsf{R}_m) \bullet (?:T)^*$$

$$\simeq \mathsf{R}_1 \bullet (?:T)^* + \mathsf{R}_2 \bullet (?:T) + \dots + \mathsf{R}_n \bullet (?:T)^*$$

چون $\mathsf{R}_k \simeq \varepsilon$ داریم:

$$\mathsf{R}_1 \bullet (?:T)^* + \mathsf{R}_2 \bullet (?:T) + \dots + \mathsf{R}_k \bullet (?:T)^* + \dots + \mathsf{R}_n \bullet (?:T)^*$$

$$\simeq \mathsf{R}_1 \bullet (?:T)^* + \mathsf{R}_2 \bullet (?:T) + \dots + \varepsilon \bullet (?:T)^* + \dots + \mathsf{R}_n \bullet (?:T)^*$$

و از طرف دیگر داریم:

$$\varepsilon \bullet (?:T)^* \simeq (?:T)^* = (\{\underline{\rho}\} \times \mathbb{S}^+)$$

پس $\mathsf{prefix}(\mathcal{S}^{\mathsf{r}}[\![\mathsf{R} \bullet (?:T)^*]\!])$ مجموعه‌ی $\{\underline{\rho}\} \times \mathbb{S}^+$ را به عنوان زیرمجموعه در درون خود دارد و عضوی بیش از این هم طبق تعریفش نمی‌تواند داشته باشد، پس:

$$\mathsf{prefix}(\mathcal{S}^{\mathsf{r}}[\![\mathsf{R} \bullet (?:T)^*]\!]) = \{\underline{\rho}\} \times \mathbb{S}^+$$

که این گزاره نتیجه می‌دهد:

$$\langle \underline{\rho}, \pi \rangle \in \mathsf{prefix}(\mathcal{S}^{\mathsf{r}}[\![\mathsf{R} \bullet (?:T)^*]\!])$$

$\blacktriangleright\ \mathsf{R}_k \not\cong \varepsilon:$

این فرض، همان طور که پیش‌تر اشاره کردیم، یعنی $\mathsf{R}_k \in \mathsf{R}^+ \cap \mathsf{R}^\dagger$. پس مجاز هستیم از تابع سر و دم استفاده کنیم. فرض می‌کنیم $\mathsf{fstnxt}(\mathsf{R}_k) = \langle \mathsf{L}_k^1 : \mathsf{B}_k^1, \mathsf{R}_k^1 \rangle$. همین‌طور فرض می‌کنیم:

$$\pi = \langle l_0, \rho_0 \rangle \langle l_1, \rho_1 \rangle \langle l_2, \rho_2 \rangle ... \langle l_l, \rho_l \rangle$$

و تعریف می‌کنیم:

$$\pi(i) = \langle l_i, \rho_i \rangle \langle l_{i+1}, \rho_{i+1} \rangle, ..., \langle l_l, \rho_l \rangle$$

داریم:

$$\mathcal{M}^t \langle \underline{\rho}, \mathsf{R}_k \rangle \pi = \langle T, \mathsf{R}' \rangle \Rightarrow \forall \mathsf{R}'' \in \mathbb{R} : \mathcal{M}^t \langle \underline{\rho}, \mathsf{R}_k \rangle \pi \neq \langle F, \mathsf{R}'' \rangle$$

پس لاجرم تساوی زیر برقرار است(با توجه به سر و دم R_k):

$$\mathcal{M}^t \langle \underline{\rho}, \mathsf{R}_k \rangle \pi = \mathcal{M}^t \langle \underline{\rho}, \mathsf{R}_k^1 \rangle \pi(1)$$

بدون کاستن از کلیت(چون ممکن است $\mathsf{R}_k^1 \cong \varepsilon$) فرض می‌کنیم کاری که انجام دادیم را می‌توانیم روی دم خروجی عبارت منظم R_k(یعنی R_k^1) تکرار کنیم:

$$\mathcal{M}^t \langle \underline{\rho}, \mathsf{R}_k^1 \rangle \pi(1) = \mathcal{M}^t \langle \underline{\rho}, \mathsf{R}_k^2 \rangle \pi(2) \quad \text{where } \mathsf{fstnxt}(\mathsf{R}_k^1) = \langle \mathsf{L}_k^2 : \mathsf{B}_k^2, \mathsf{R}_k^2 \rangle$$

باز هم بدون کاستن از کلیت می‌توانیم فرض کنیم که این رویه را به صورت یک سلسله می‌توان تا h مرحله ادامه داد، یعنی:

$$\mathcal{M}^t \langle \underline{\rho}, \mathsf{R}_k^{h-1} \rangle \pi(h-1) = \mathcal{M}^t \langle \underline{\rho}, \mathsf{R}_k^h \rangle \pi(h) \quad \text{where } \mathsf{fstnxt}(\mathsf{R}_k^{h-1}) = \langle \mathsf{L}_k^h : \mathsf{B}_k^h, \mathsf{R}_k^h \rangle$$

در حالیکه $\mathsf{R}_\mathsf{k}^\mathsf{h} \cong \varepsilon$. اگر $\mathsf{R}_\mathsf{k}^\mathsf{h} \cong \varepsilon$ هیچ زمانی برقرار نشود، یعنی بتوانیم این سلسله را تا بی‌نهایت ادامه دهیم، مطمئن خواهیم بود که در معنای R_k حتما ردهای پیشوندی نامتناهی حضور دارند

که چنین چیزی با تعریف معنای عبارات منظم در تناقض است، چون در معنای عبارات منظم رد پیشوندی نامتناهی حضور ندارد. تا اینجا، می‌توانیم بگوییم:

$$\mathsf{R_k} \doteqdot \mathsf{L_k^1 : B_k^1 \bullet L_k^2 : B_k^2 \bullet ... \bullet L_k^h : B_k^h}$$

حال بسته به اینکه $h < l$ برقرار هست یا خیر می‌توانیم مسئله را به دو حالت افراز کنیم:

$$\blacktriangleright\blacktriangleright\ h < l :$$

در این صورت داریم:

$$\mathcal{M}^t\langle \underline{\rho}, \mathsf{R}_k^h\rangle\pi(h+1) = \langle T, \varepsilon\rangle$$

این‌ها یعنی داریم:

$$\forall j : 1 \leq j \leq h \rightarrow \langle \underline{\rho}, \langle l_j, \rho_j\rangle\rangle \in \mathcal{S}^r[\![\mathsf{L}_j : \mathsf{B}_j]\!]$$

$$\Rightarrow \langle \underline{\rho}, \pi\rangle \in \mathcal{S}^r[\![\mathsf{R}_k \bullet (? : T)^*]\!] \Rightarrow \langle \underline{\rho}, \pi\rangle \in \mathsf{prefix}(\mathcal{S}^r[\![\mathsf{R} \bullet (? : T)^*]\!])$$

$$\blacktriangleright\blacktriangleright\ h \geq l :$$

در این صورت داریم:

$$\mathcal{M}^t\langle \underline{\rho}, \mathsf{R_k}\rangle\pi = \langle T, \mathsf{R}_k^l\rangle$$

که یعنی:

$$\forall j : 1 \leq j \leq l \rightarrow \langle \underline{\rho}, \langle l_j, \rho_j\rangle\rangle \in \mathcal{S}^r[\![\mathsf{L}_j : \mathsf{B}_j]\!]$$

$$\Rightarrow \langle \underline{\rho}, \pi\rangle \in \mathsf{prefix}(\mathcal{S}^r[\![\mathsf{R}_k]\!]) \Rightarrow \langle \underline{\rho}, \pi\rangle \in \mathsf{prefix}(\mathcal{S}^r[\![\mathsf{R} \bullet (? : T)^*]\!])$$

پس در کل می‌توانیم بگوییم

$$\langle \underline{\rho}, \pi\rangle \in \mathsf{prefix}(\mathcal{S}^r[\![\mathsf{R} \bullet (? : T)^*]\!])$$

و اثبات قضیه تمام می‌شود. □

حال به اثبات تمامیت می‌پردازیم.

قضیه ۱۶.۴. (قضیه تمامیت): اگر P یک برنامه، R یک عبارت منظم و $\underline{\rho}$ یک محیط اولیه باشند، آنگاه داریم:

$$\mathsf{P}, \underline{\rho} \models \mathsf{R} \Rightarrow \mathsf{P}, \underline{\rho} \models_r \mathsf{R}$$

اثبات. با برهان خلف این قضیه را ثابت می‌کنیم و شکل اثبات تا حدی شبیه به اثبات درستی است.

$$\{\underline{\rho}\}\times\mathcal{S}^*[\![\mathsf{P}]\!] \nsubseteq \mathcal{M}\langle\underline{\rho},\mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!] \Rightarrow \exists\pi : \langle\underline{\rho},\pi\rangle \in \{\rho\}\times\mathcal{S}^*[\![\mathsf{P}]\!]\wedge\langle\underline{\rho},\pi\rangle \notin \mathcal{M}\langle\underline{\rho},\mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!]$$

اگر فرض کنیم $\mathsf{dnf(R)} = \mathsf{R_1} + \mathsf{R_2} + ... + \mathsf{R_n}$، علاوه بر این، با توجه به آنچه در اثبات درستی گفتیم فرض کنیم:

$$\mathsf{R}_i \Bumpeq \mathsf{L_i^1} : \mathsf{B_i^1} \bullet \mathsf{L_i^2} : \mathsf{B_i^2} \bullet ... \bullet \mathsf{L_i^n} : \mathsf{B_i^n}$$

و

$$\pi = \langle l_1,\rho_1\rangle\langle l_2,\rho_2\rangle...\langle l_l,\rho_l\rangle$$

می‌توانیم در ادامه‌ی فرض خلف نتیجه بگیریم:

$$\forall i : 1 \leq i \leq n \rightarrow \langle\underline{\rho},\pi\rangle \notin \mathcal{M}^{\dagger}\langle\underline{\rho},\mathsf{R}_i\rangle\mathcal{S}^*[\![\mathsf{P}]\!]$$

$$\Rightarrow \forall i : 1 \leq i \leq n \rightarrow \exists\mathsf{R'}_i :\Rightarrow \mathcal{M}^t\langle\underline{\rho},\mathsf{R}_i\rangle\pi = \langle F,\mathsf{R}_i^k\rangle$$

در این صورت خواهیم داشت:

$$\exists j : \langle\underline{\rho},\langle l_j,\rho_j\rangle\rangle \notin \mathcal{S}^r[\![\mathsf{L_i^j} : \mathsf{B_i^j}]\!] \Rightarrow \langle\underline{\rho},\pi\rangle \notin \mathcal{S}^r[\![\mathsf{R}_i \bullet (?:T)^*]\!]$$

از نتیجه‌ی آخر می‌توانیم ثابت کنیم $\langle\underline{\rho},\pi\rangle \notin \mathsf{prefix}(\mathcal{S}^r[\![\mathsf{R}_i \bullet (?:T)^*]\!])$ چون اگر غیر از این باشد، اگر فرض کنیم $\langle\underline{\rho},\pi'\rangle$ عضو $\mathsf{prefix}(\mathcal{S}^r[\![\mathsf{R}_i \bullet (?:T)^*]\!])$ هست، اما عضو $\mathcal{S}^r[\![\mathsf{R}_i \bullet (?:T)^*]\!]$ نیست، اگر طول π بزرگتر یا مساوی j باشد، به خاطر وجود $\langle l_j,\rho_j\rangle$ در π، خواهیم داشت $\pi' \neq \pi$ و اگر طول π' کمتر از j باشد، چون طول π قطعا بزرگتر یا مساوی j است(نتیجه از عبارت بالایی که دارای سور وجودی است)، پس باز هم $\pi' \neq \pi$. توجه شود که $\langle\underline{\rho},\pi\rangle \notin \mathcal{S}^r[\![\mathsf{R}_i \bullet (?:T)^*]\!]$ با توجه به آنچه گفتیم، با فرض در تناقض است و حکم ثابت می‌شود. □

۳.۴ در مورد قدرت بیان عبارات منظم

در این بخش می‌خواهیم کمی در مورد قدرت بیان عبارات منظمی که در این فصل آورده‌ایم در مقایسه با منطق LTL که در فصل اول آمده صحبت کنیم. همان طور که پیش‌تر گفتیم، یکی از دلایلی که کوزو برای استفاده از عبارات منظم دارد این است که عبارات منظم قادر به بیان خواصی هستند که منطق زمانی از بیان آن‌ها عاجز است. او [۲۵] را به عنوان مرجع صحبتش در نظر گرفته و در [۶] در این مورد صحبت بیشتری نکرده است. در این بخش می‌خواهیم این بحث را بیشتر باز کنیم.

سوال اصلی ما در این بخش این است که آیا یکی از این دو موجود، اکیدا از دیگری در بیان قوی‌تر هست یا خیر. به این معنی که آیا می‌شود هر چیزی که با یکی از این‌ها قابل بیان است را با دیگری هم بیان کرد یا خیر. البته [۲۵] حداقل در مورد یک طرف این بحث حرف زده و ما هم در اینجا از آن محتوا هم کمک خواهیم گرفت و این بحث را میان دو زبانی که تا به حال در بحث داشتیم مطرح می‌کنیم، یعنی منطق زمانی خطی و عبارات منظمی که در همین فصل معرفی شدند.

۱.۳.۴ نزدیک کردن صورت دو زبان

پیش از اینکه بخواهیم مقایسه‌ای ترتیب دهیم، ابتدا باید زبانی را که در فصل اول از LTL آورده‌ایم، با عبارات منظمی که در این فصل آورده‌ایم با هم قابل مقایسه کنیم. به هر حال زبانی که در فصل اول آمده یک منطق گزاره‌ای است اما در عبارات منظمی که در این فصل آورده‌ایم اتم‌ها(یا به عبارت دیگر لیترال‌ها) به موجودات ساختارمندتری تبدیل شده‌اند که همان زوج مرتب‌های $\mathsf{L} : \mathsf{B}$ هستند. در اینجا معناشناسی ما هم نسبت به تغییری که در اتم‌ها داده‌ایم فرق کرده است. بنابراین اولین تلاشی که می‌کنیم این است که منطقی که در فصل اول آورده‌ایم را به شکلی که حس می‌کنیم قابل مقایسه با عبارات منظم باشد تغییر دهیم. در واقع تغییری که در زبان می‌دهیم همان تغییر اتم‌هاست. در ادامه معناشناسی منطق LTL را هم به کمک ردهای پیشوندی، مثل عبارات منظم، بیان می‌کنیم. نام این منطق جدید را "LTLـ گسترش یافته" گذاشته‌ایم.

تعریف ۱۷.۴. (زبان LTLـ گسترش یافته): با فرض اینکه $\mathsf{L} \subseteq \mathbb{L}$ و $\mathsf{B} \in \mathbb{B}$ زبان LTLـ گسترش یافته به شکل زیر است:

$$\phi \in \Phi \Leftrightarrow \phi ::= \mathsf{L} : \mathsf{B} | \phi_1 \vee \phi_2 | \neg\phi_1 | \bigcirc \phi_1 | \phi_1 \mathcal{U} \phi_2$$

مجموعه‌ی همه‌ی فرمول‌ها در این زبان را با $\mathfrak{L}_e$ نمایش می‌دهیم.

تعریف ۱۸.۴. (معناشناسی LTLـ گسترش یافته): تابع $\mathcal{S}^t : \Phi \to P(\underline{\mathbb{EV}} \times \mathfrak{S}^+)$ با ضابطه‌ی زیر، معناشناسی زبان LTL است.

$$\blacktriangleleft\ \mathcal{S}^t[\![\mathsf{L} : \mathsf{B}]\!] = \{\langle \underline{\rho}, \langle l, \rho\rangle\rangle | l \in \mathsf{L}\ ,\ \mathcal{B}[\![\mathsf{B}]\!]\underline{\rho}, \rho = T\ ,\ \underline{\rho} \in \underline{\mathbb{EV}}\}$$

$$\blacktriangleleft\ \mathcal{S}^t[\![\phi_1 \vee \phi_2]\!] = \mathcal{S}^t[\![\phi_1]\!] \cup \mathcal{S}^t[\![\phi_2]\!]$$

$$\blacktriangleleft\ \mathcal{S}^t[\![\neg\phi_1]\!] = \mathfrak{S}^+ \setminus \mathcal{S}^t[\![\phi_1]\!]$$

$$\blacktriangleleft\ \mathcal{S}^t[\![\bigcirc\phi_1]\!] = \{\langle \underline{\rho}, \langle l, \rho\rangle\pi\rangle | \langle \underline{\rho}, \pi\rangle \in \mathcal{S}^t[\![\phi_1]\!]\ ,\ l \in \mathbb{L}\ ,\ \rho \in \mathbb{EV}\ ,\ \underline{\rho}, \in \underline{\mathbb{EV}}\}$$

$$\blacktriangleleft\ \mathcal{S}^t[\![\phi_1 \mathcal{U} \phi_2]\!] = \{\langle \underline{\rho}, \pi\rangle | \exists \pi' : \forall \pi'' : \pi\pi'' \subsetneq \pi' \to (\langle \underline{\rho}, \pi\pi''\rangle \in \mathcal{S}^t[\![\phi_1]\!], \langle \underline{\rho}, \pi'\rangle \in \mathcal{S}^t[\![\phi_2]\!])\ ,\ \underline{\rho} \in \underline{\mathbb{EV}}\}$$

حال می‌خواهیم ثابت کنیم که معناشناسی‌ای که ارائه کرده‌ایم، معنای منطق LTL را حفظ کرده. برای این کار ابتدا یک معناشناسی جدید را برای زبان LTL قدیمی ارائه می‌دهیم و ثابت می‌کنیم که معادل با معناشناسی قبلی است.

تعریف ۱۹.۴. (مدل LTLـ جدید): به هر تابع از Π یعنی مجموعه‌ی اتم‌ها به $P(\mathbb{N})$ یعنی مجموعه‌ی همه‌ی زیرمجموعه‌های مجموعه‌ی کل اعداد طبیعی می‌گوییم مدل جدید.

$$M_n : \Pi \to P(\mathbb{N})$$

مجموعه‌ی همه‌ی مدل‌های جدید را با $\mathbb{M}_n$ نشان می‌دهیم.

تعریف ۲۰.۴. (معناشناسی LTLـ جدید): تابع $\bar{\ } : \mathbb{M}_n \to \Phi \to P(\mathbb{N})$ (به علامت گذاری دقت کنید! نام گذاری تابع به شکل سنتی و با یک حرف خاص نیست و صرفا علامت بار را گذاشته‌ایم. به ازای مدل M_n در ورودی، تابع $\bar{M_n} : \Phi \to P(\mathbb{N})$ را داریم.) به ازای مدل M_n روی فرمول‌های زبان LTL به شکل زیر تعریف می‌شود:

$$◀\ \bar{M_n}(\pi) = M_n(\pi)$$

$$◀\ \bar{M_n}(\phi \vee \psi) = \bar{M_n}(\phi) \cup \bar{M_n}(\psi)$$

$$◀\ \bar{M_n}(\neg\phi) = \mathbb{N} \setminus \bar{M_n}(\phi)$$

$$◀\ \bar{M_n}(\bigcirc\phi) = \{n+1 | n \in \bar{M_n}(\phi)\}$$

$$◀\ \bar{M_n}(\phi \mathcal{U} \psi) = \{n | \exists k : \forall j : n \leq j \lneq k \to j \in \bar{M_n}(\phi), k \in \bar{M_n}(\psi)\}$$

و به کمک این تابع تعریف می‌کنیم:

$$M_n, i \models_n \phi \ \mathit{iff} \ i \in \bar{M_n}(\phi)$$

مدل‌های جدیدی که تعریف کرده‌ایم همان اطلاعاتی را که مدل‌های قدیمی به ما می‌دادند، با آرایش دیگری در خود نگه می‌دارند. برای اینکه بتوانیم از هر دو شیوه‌ی بیان یک مدل استفاده کنیم یک تابع میان آن‌ها تعریف می‌کنیم.

تعریف ۲۱.۴. (تابع مبدل): تابع $\mathfrak{T} : \mathbb{M}_n \to \mathbb{M}$ را به نام تابع مبدل به شکل زیر تعریف می‌کنیم:

$$\mathfrak{T}(M_n)(i) = \{\pi | i \in M_n(\pi)\}$$

این تابع یک تناظر یک به یک و پوشا بین مدل‌های قدیم و جدید است.

قضیه ۲۲.۴. تابع $\mathfrak{T}$ یک به یک و پوشا است.

اثبات. پیش از هر چیز ذکر این نکته ضروری است که اصلا چرا $\mathfrak{T}$ یک تابع است. این از این می‌آید که با توجه به اینکه در تعریف این تابع صرفا از عملگر اجتماع و محمول عضویت استفاده شده و می‌دانیم این دو خوش تعریف هستند، پس متوجه می‌شویم که $\mathfrak{T}$ یک تابع است.

اثبات یک به یک بودن: فرض می‌کنیم که به ازای دو مدل M_n و M'_n که ممکن است متفاوت باشند، داریم $\mathfrak{T}(M_n) = \mathfrak{T}(M'_n)$. داریم:

$$\Rightarrow \forall i \in \mathbb{N} : \mathfrak{T}(M_n)(i) = \mathfrak{T}(M'_n)(i) \Rightarrow \forall i \in \mathbb{N} : \forall \pi \in \Pi : \pi \in \mathfrak{T}(M_n)(i) \leftrightarrow \pi \in \mathfrak{T}(M'_n)(i)$$

$$\Rightarrow \forall i \in \mathbb{N} : \forall \pi \in \Pi : i \in M_n(\pi) \leftrightarrow \pi \in M'_n(\pi) \iff M_n = M'_n$$

پس این دو مدل الزاما برابرند و این یعنی این تابع یک به یک است.

اثبات پوشا بودن: فرض می‌کنیم $M \in \mathbb{M}$ و ثابت می‌کنیم برای مدل $M_n(\pi) = \{i | \pi \in M(i)\}$ داریم $\mathfrak{T}(M_n) = M$.

$$\forall j \in \mathbb{N} : \mathfrak{T}(M_n)(j) = \{\pi | j \in M_n(\pi)\} = \{\pi | j \in \{i | \pi \in M(i)\}\} = \{\pi | \pi \in M(j)\} = M(j)$$

$$\Rightarrow \mathfrak{T}(M_n) = M$$

پس تابع مبدل پوشا نیز هست. □

بنابراین می‌توانیم قضیه‌ی زیر را بیان کنیم.

قضیه ۲۳.۴. معناشناسی جدید و قدیم با یکدیگر معادل‌اند یا به عبارت دیگر برای هر مدل $M_n \in \mathbb{M}_n$ داریم:

$$\forall \phi \in \Phi, i \in \mathbb{N} : \bar{M}_n, i \models_n \phi \leftrightarrow \mathfrak{T}(M_n), i \models \phi$$

اثبات. روی ساختار ϕ استقرا می‌زنیم:

$$\blacktriangleright\ \phi = \pi :$$

$$M_n, i \models_n \pi \iff i \in \bar{M}_n(\pi) \iff i \in M_n(\pi)$$
$$\iff \pi \in \mathfrak{T}(M_n)(i) \iff \mathfrak{T}(M_n), i \models \pi$$

$$\blacktriangleright\ \phi = \phi \vee \psi :$$

$$M_n, i \models_n \phi \vee \psi \iff i \in \bar{M}_n(\phi \vee \psi) = \bar{M}_n(\phi) \cup \bar{M}_n(\psi)$$

در اینجا بدون کاستن از کلیت می‌توانیم فرض کنیم $i \in \bar{M}_n$. با فرض دیگر هم اثبات به همین شکل است.

$$i \in \bar{M}_n(\phi) \iff M_n, i \models_n \phi \mathfrak{T}(M_n), i \models \phi \Rightarrow \mathfrak{T}(M_n), i \models \phi \vee \psi$$

عکس این اثبات را هم برای عکس این طرف قضیه که ثابت کردیم در همین اثبات می‌شود دید. آخرین نتیجه‌ای که گرفتیم می‌تواند بدون کاستن از کلیت برعکس گرفته شود و در واقع برای هر دو زیر فرمول به‌طور جداگانه ثابت شود.

$$\blacktriangleright\ \phi = \neg\phi :$$

$$M_n, i \models_n \iff i \in \bar{M_n}(\neg\phi) \iff i \notin \bar{M_n}(\phi)$$
$$\iff M_n, i \not\models_n \phi \iff \mathfrak{T}(M_n), i \not\models \phi \iff \mathfrak{T}(M_n), i \models \neg\phi$$

$$\blacktriangleright\ \phi = \phi\mathcal{U}\psi :$$
$$M_n, i \models_n \pi\mathcal{U}\psi \iff i \in \bar{M_n}(\phi\mathcal{U}\psi)$$
$$\iff \exists k : \forall j : i \leq j < k \to j \in \bar{M_n}(\phi), k \in \bar{M_n}(\psi)$$
$$\iff \exists k : \forall j : i \leq j < k \to M_n, j \models_n \phi, M_n, k \models_n \psi$$
$$\iff \exists k : \forall j : i \leq j < k \to \mathfrak{T}(M_n), j \models \phi, \mathfrak{T}(M_n), k \models \psi$$
$$\iff \mathfrak{T}(M_n), i \models \phi\mathcal{U}\psi$$

$\square$

پس تا اینجای کار تا حدودی نشان دادم که معناشناسی جدیدی که برای LTL ارائه کرده‌ایم با معناشناسی قدیمی‌اش معادل است.

می‌توان دید که $\bar{M_n}$ به S^t بسیار شبیه است و انگار که با جایگذاری زوج‌های L : B به جای اتم‌ها و ردهای پیشوندی متناهی به جای اعداد طبیعی می‌توان از $\bar{M_n}$ به S^t رسید. البته می‌توان برای اطمینان به مشخص‌تر کردن این ارتباط ادامه داد. در واقع ماهیت مدل‌ها در دو منطقی که ادعای معادل بودنشان را داریم، متفاوت است. در منطق قبلی مدل ها هر اتم را به زیرمجموعه‌ای از همه‌ی اعداد طبیعی می‌نگارند و در منطق گسترش یافته مدل‌ها هر اتم را به مجموعه‌ای از وضعیت‌ها(ردهای پیشوندی تک عضوی) می‌نگارند. این یعنی ترتیبی که در ردهای پیشوندی داریم در مدل وجود ندارد، درحالیکه در مدل‌های قدیمی ترتیب از مدل مشخص می‌شود. از آنجایی که در هر دو منطق مجموعه‌ی مدل‌ها هم مرتبه‌ی مجموعه‌ی اعداد حقیقی است، می‌دانیم که یک دو سویی بین مدل‌ها هست و احتمالا صورت هم‌ارزی دو منطق را می‌توان با استفاده از آن تابع و البته دوسویی دیگری بین اتم‌های دو منطق، که از شمارا و نامتناهی بودن مجموعه‌ی اتم‌های هر دو منطق داریم، می‌توان به یک صورت کامل برای این ادعا هم رسید. اما ما فعلا به همین شواهد قانع شده‌ایم.

۲.۳.۴ مقایسه

حال به بررسی این می‌پردازیم که کدام زبان قدرت بیان بیشتری دارد. ابتدا ثابت می‌کنیم عبارات منظمی وجود دارند که قابل بیان به وسیله‌ی LTL- گسترش یافته نیستند. سپس عکس این را ثابت می‌کنیم، یعنی ثابت نشان می‌دهیم فرمولی در LTL- گسترش یافته وجود دارد که به وسیله‌ی عبارات منظم قابل بیان نیست.

قضیه ۲۴.۴. به ازای $\mathsf{L} \subseteq \mathbb{L}, \mathsf{B} \in \mathbb{B}$ عبارت منظم $(\langle ?:T\rangle\langle \mathsf{L}:\mathsf{B}\rangle)^*$ قابل بیان با LTL نیست. به عبارت دیگر هیچ فرمولی در LTL- گسترش یافته نیست که هم‌معنا با عبارت منظم $(\langle ?:T\rangle\langle \mathsf{L}:\mathsf{B}\rangle)^*$ باشد.

اثبات. اثبات را با استقرا روی ساختار زبان LTL- گسترش یافته انجام می‌دهیم. در هر مورد از استقرا ثابت می‌کنیم که اگر ساختار فرمول به شکل فرض شده باشد، نمی‌تواند هم معنا با عبارت منظم ذکر شده باشد.

$$\blacktriangleright\ \phi = \langle \mathsf{L}:\mathsf{B}\rangle :$$

در این حالت معنای ϕ صرفا شامل ردهای پیشوندی تک عضوی است در حالی که در معنای عبارت منظم مذکور اصلا رد پیشوندی تک عضوی وجود ندارد.

$$\blacktriangleright\ \phi = \phi_1 \vee \phi_2 :$$

$$\blacktriangleright\ \phi = \neg\phi_1 :$$

□

فصل ۵

وارسی مدل ساختارمند

در این فصل همان طور که پیشتر هم بارها اشاره کردیم قرار است، به ادامه‌ی ساختارمندتر کردن کار بپردازیم. در فصل گذشته ساختار عبارات منظم را به تعریف وارسی مدل اضافه کردیم و حالا می‌خواهیم ساختار زبانمان را به کار اضافه کنیم. این آخرین تلاش [۶] برای گسترش کار بوده. یعنی وارسی مدل به شکل جدید تعریف شده و معادل بودن آن با صورت قبلی وارسی مدل ثابت شده و پس از آن کار پایان می‌پذیرد.

تعریف صورت جدید چونکه روی ساختار زبان انجام گرفته جزئیات بسیار طولانی‌ای دارد. همی باعث شده تا اثبات برابری این صورت با صورت قبلی هم بسیار مفصل و حجیم باشد. این اثبات در [۶] به طور کامل حین معرفی هر مورد تعریف بیان شده. بنابراین از ارائه‌ی دوباره‌ی این جزئیات خودداری کرده‌ایم و صرفا ممکن است برای بعضی موارد آن یک گزارش کلی از اینکه هر برقراری چگونه ثابت شده، به سبک [۶] بعد از بیان هر مورد از تعریف، آورده‌باشیم.

تعریف ۱.۵ . تابع $\hat{\mathcal{M}}$ را از نوع $(\underline{\mathbb{EV}} \times \mathbb{R}) \to P(\mathfrak{S}^{+\infty}) \to (\underline{\mathbb{EV}} \times P(\mathfrak{S}^{+\infty}))$ وارسی مدل ساختارمند می‌نامیم(ضابطه‌ی تابع در ادامه‌ی متن آمده).

در ادامه ممکن است به خاطر کوتاه‌تر نوشتن بعضی جاها به‌جای $\hat{\mathcal{M}}\langle \underline{\rho}, \mathsf{R}\rangle \mathcal{S}^*[\![\mathsf{P}]\!]$ از $\hat{\mathcal{M}}\langle \underline{\rho}, \mathsf{R}\rangle [\![\mathsf{P}]\!]$ استفاده کرده باشیم، یعنی در اشاره به تابع $\mathcal{S}^*$ به براکت‌ها $[\![\ ]\!]$ قناعت کرده باشیم.

تعریف روی ساختار مجموعه‌ی $\mathbb{P} \cup \mathbb{SI} \cup \mathbb{S}$ انجام شده. یعنی در واقع روی همه‌ی اعضای تعریف هر یک از این سه بخش زبان تعریف کردن صورت گرفته، تقریبا کاری شبیه به اثبات لمی که در بخش ۳.۳ داشتیم. در ادامه موردهای تعریف $\hat{\mathcal{M}}$ را به ازای برنامه‌ی P، محیط اولیه‌ی $\underline{\rho}$ و عبارت منظم Rتعریف می‌کنیم. یعنی در حال تعریف $\hat{\mathcal{M}}\langle \underline{\rho}, \mathsf{R}\rangle \mathcal{S}^*[\![\mathsf{P}]\!]$ هستیم، روی ساختار برنامه‌ها یعنی P.

$$\blacktriangleleft\ \mathsf{P} = \mathsf{SI}:$$

$$\hat{\mathcal{M}}\langle \underline{\rho}, \mathsf{R}\rangle \mathcal{S}^*[\![\mathsf{P}]\!] = \bigcup_{i=1}^{n} \{\langle \underline{\rho}, \pi\rangle | \exists \mathsf{R}' \in \mathbb{R},\ \langle \pi, \mathsf{R}'\rangle \hat{\mathcal{M}}^{\dagger}\langle \underline{\rho}, \mathsf{R}_i\rangle \mathcal{S}^*[\![\mathsf{SI}]\!]\}$$

$$\text{where } \mathsf{dnf}(\mathsf{R}) = \mathsf{R}_1 + \mathsf{R}_2 + \ldots + \mathsf{R}_n$$

اثبات برابری با صورت فصل قبل با اینکه $\hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{Sl}]\!]$ هنوز تعریف نشده در [۶] آمده با این استدلال که برابری $\hat{\mathcal{M}}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{Sl}]\!] = \mathcal{M}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{Sl}]\!]$ در فرض استقرا آمده. کلیت اثبات هم این است که از باز کردن تعریف $\mathcal{M}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{P}]\!]$ با استفاده‌ی مستقیم تعاریف و بدون تکنیک خاصی به $\hat{\mathcal{M}}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{P}]\!]$ رسیده.

در ادامه با توجه به تعریف قبل به بیان تعریف $\hat{\mathcal{M}}^{\dagger}$ پرداخته شده. این تنها بخش تابع $\hat{\mathcal{M}}$ است که معرفی نشده و با مشخص شدن آن معنای $\hat{\mathcal{M}}$ به ازای برنامه‌های مختلف مشخص می‌شود. این نکته را در نظر داشته باشیم که $\hat{\mathcal{M}}^{\dagger}$ قرار است در عمل روی مجموعه‌ی برنامه‌ها تعریف شود و مثلا اینکه به ازای $\Pi \in \mathfrak{S}^{+\infty}$ دلخواه که مساوی معنی یک برنامه نیست، این تابع با یک محیط اولیه و یک عبارت منظم چه خروجی‌ای دارد برای ما اهمیتی ندارد و البته به این شکلی هم که تابع تعریف شده حتی به ازای چنین ورودی‌ای خروجی ندارد. در واقع حتی به ازای $\mathsf{S} ::= \ \mathsf{x} \doteq \mathsf{A};$ هم که یک عضو از زبان است، به این معنا که S صرفا یک بخش از یک برنامه است و خروجی‌ای نخواهد داشت و داشتن خروجی به ازای این متغیر را از $\hat{\mathcal{M}}^{\dagger}$ انتظار داریم. مشابه $\mathcal{M}^{\dagger}$ خروجی $\hat{\mathcal{M}}^{\dagger}$ هم یک زوج مرتب شامل π ای که R را ارضا کرده و یک عبارت منظم بدون $+$ که بخشی از R را نشان می‌دهد که با π تطابق داده نشده(چون احتمالا π کوتاه‌تر بوده یا ممکن است اصلا این عبارت منظم تهی باشد).

$$\blacktriangleleft\ \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\varepsilon\rangle[\![\mathsf{S}]\!] = \{\langle\pi,\varepsilon\rangle | \pi \in \mathcal{S}^{*}]\!]\mathsf{S}[\![\}$$

برای $\mathsf{Sl} = \mathsf{SL}'\ \mathsf{S}$ و $\mathsf{R} \in \mathbb{R}^{\dagger} \cap \mathbb{R}^{+}$ داریم:

$$\blacktriangleleft\ \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{Sl}]\!] = \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{Sl}']\!]\cup$$
$$\{\langle\pi\langle\mathsf{at}[\![\mathsf{S}]\!],\rho\rangle\pi',\mathsf{R}''\rangle | \langle\pi\langle at[\![\mathsf{S}]\!],\rho\rangle,\mathsf{R}'\rangle \in \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{Sl}']\!]\wedge$$
$$\langle\langle at[\![\mathsf{S}]\!],\rho\rangle\pi',\mathsf{R}''\rangle \in \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}'\rangle[\![\mathsf{S}]\!]\}$$

از اینجا به بعد کار با تعاریف طویل‌تری از آنچه تا حالا داشتیم، روبرو هستیم، مانند همین تعریف بالا. اما به هرحال مفهوم چندان پیچیده‌ای پشت این تعاریف نیست. تعریف بالا به طور خلاصه می‌گوید همه‌ی ردهای پیشوندی ای که در $\hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{Sl}']\!]$ هستند به همراه همه‌ی زوج‌هایی که مولفه اول آن‌ها یک رد پیشوندی است که تکه‌ی اول آن داخل $\hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{Sl}']\!]$ افتاده و ادامه‌ی آن داخل $\hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}'\rangle[\![\mathsf{S}]\!]$ است، در حالیکه R' هم ادامه‌ی عبارت منظم R است، که به تطبیق با رد پیشوندی داخل $\hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{Sl}']\!]$ که زوج خودش است نرسیده و مولفه‌ی دوم این زوج مرتب نیز ادامه‌ی عبارت منظم R است که به تطبیق با رد پیشوندی‌ای که در مولفه‌ی اول بود، نرسیده.

برای $\mathsf{Sl} = \epsilon$ و $\mathsf{R} \in \mathbb{R}^{\dagger} \cap \mathbb{R}^{+}$ داریم:

$$\blacktriangleleft\ \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{Sl}]\!] =$$
$$\{\langle\langle at[\![\mathsf{Sl}]\!],\rho\rangle,\mathsf{R}'\rangle|\langle\underline{\rho},\langle at[\![\mathsf{Sl}]\!],\rho\rangle\rangle\in\mathcal{S}^{r}[\![\mathsf{L}:\mathsf{B}]\!]\}$$
$$\text{where } \mathsf{fstnxt}(\mathsf{R}) = \langle\mathsf{L}:\mathsf{B},\mathsf{R}'\rangle$$

یعنی همه‌ی ردهای پیشوندی تک عضوی که محیطشان اولین لبترال موجود در عبارت منظم را ارضا می‌کند. در واقع هر مخیطی که این لیترال را ارضا کند، برچسب این مجوعه دستور را در این مجموعه می‌آورد(به همراه باقی عبارت منظم).

برای $\mathsf{S} =\ \mathsf{x} \doteq \mathsf{A};$ و $\mathsf{R}\in\mathbb{R}^{\dagger}\cap\mathbb{R}^{+}$ داریم:

$$\blacktriangleleft\ \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{S}]\!] =$$
$$\{\langle\langle at[\![\mathsf{S}]\!],\rho\rangle,\mathsf{R}'\rangle|\langle\underline{\rho},\langle at[\![\mathsf{S}]\!],\rho\rangle\rangle\in\mathcal{S}^{r}[\![\mathsf{L}:\mathsf{B}]\!]\}$$
$$\cup\{\langle\langle at[\![\mathsf{S}]\!],\rho\rangle\langle aft[\![\mathsf{S}]\!],\rho[\mathsf{x}\leftarrow\mathcal{A}[\![\mathsf{A}]\!]\rho]\rangle,\varepsilon\rangle|\mathsf{R}'\in\mathbb{R}_{\varepsilon}\wedge$$
$$\langle\underline{\rho},\langle at[\![\mathsf{S}]\!],\rho\rangle\rangle\in\mathcal{S}^{r}[\![\mathsf{L}:\mathsf{B}]\!]\}$$
$$\cup\{\langle\langle at[\![\mathsf{S}]\!],\rho\rangle\langle aft[\![\mathsf{S}]\!],\rho[\mathsf{x}\leftarrow[\![\mathsf{A}]\!]\rho]\rangle,\mathsf{R}''\rangle|\mathsf{R}'\notin\mathbb{R}_{\varepsilon}\wedge$$
$$\langle\underline{\rho},\langle at[\![\mathsf{S}]\!],\rho\rangle\rangle\in\mathcal{S}^{r}[\![\mathsf{L}:\mathsf{B}]\!]\wedge\langle\mathsf{L}':\mathsf{B}',\mathsf{R}''\rangle=\mathsf{fstnxt}(\mathsf{R}')\wedge$$
$$\langle\underline{\rho},\langle aft[\![\mathsf{S}]\!],\rho[\mathsf{x}\leftarrow\mathcal{A}[\![\mathsf{A}]\!]\rho]\rangle\rangle\in\mathcal{S}^{r}[\![\mathsf{L}':\mathsf{B}']\!]\}$$

با نگاه به این تعریف می‌توان بهتر متوجه شد که اینکه می‌گفتیم قرار است ساختار زبان وارد صورت فصل پیش شود یعنی چه. به جای اینکه مانند فصل گذشته نسبت به ردهای پیشوندی متفاوت خروجی تابع تعیین شود، نسبت به دستوری که معنایش ردهای پیشوندی هستند، خروجی تعیین می‌شود.

در اینجا هم با اینکه تعریف متکلف است اما معنای ساده‌ای دارد. این تابع دستور را به همراه زوج محیط اولیه و عبارت منظم می‌گیرد، ابتدا همان چیزهایی را که تابع در حالت قبلی برمی‌گرداند را برمی‌گرداند و سپس نسبت به اینکه پس از تغییر در محیط‌ها در اثر اجرای دستور مقدار دهی ادامه‌ی عبارت منظم سارگار هست یا نه زوج‌های متشکل از رد پیشوندی و عبارت منظم را به خروجی اضافه می‌کند. نکته‌ای که به جزئیات این تعریف اضافه کرده این است که برای اینکه ادامه‌ی عبارات منظم خارج شده بعد از بررسی اولین لیترال عبارت منظم، یعنی R'، آیا تهی است یا نه و این دو حالت متفاوت را در بر خواهد گرفت که در تعریف لحاظ شده‌اند.

از این ۴ حالت تنها اثبات حالت آخر در [۶] آورده شده و خب اثبات دیگر حالات را هم می‌تواند در همین اثبات که مفصل‌تر و کلی‌تر است، دید. اثبات سر راست است، از جایگذاری تساوی‌های واضح استفاده شده و جزئیات و توضیحات کافی دارد.

برای عبارت منظم $\mathsf{R} \in \mathbb{R}^{\dagger} \cup \mathbb{R}^{+}$ و $\mathsf{S} = \ \mathsf{if}\ (\mathsf{B})\ \mathsf{S_t}$ داریم:

$$\blacktriangleleft\ \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{S}]\!] =$$
$$\{\langle\langle at[\![\mathsf{S}]\!],\rho\rangle,\mathsf{R}'\rangle|\langle\underline{\rho},\langle at[\![\mathsf{S}]\!],\rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L}' : \mathsf{B}']\!]\}$$
$$\cup\{\langle\langle at[\![\mathsf{S}]\!],\rho\rangle\langle at[\![\mathsf{S_t}]\!],\rho\rangle\pi,\mathsf{R}''|\mathcal{B}[\![\mathsf{B}]\!]\rho = T\wedge$$
$$\langle\underline{\rho},\langle at[\![\mathsf{S}]\!],\rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L}' : \mathsf{B}']\!]\wedge$$
$$\langle\langle at[\![\mathsf{S_t}]\!],\rho\rangle\pi,\mathsf{R}''\rangle \in \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}'\rangle[\![\mathsf{S_t}]\!]\}$$
$$\cup\{\langle\langle at[\![\mathsf{S}]\!],\rho\rangle\langle aft[\![\mathsf{S}]\!],\rho\rangle,\varepsilon\rangle|\mathcal{B}[\![\mathsf{B}]\!]\rho = F\wedge \mathsf{R}' \in \mathbb{R}_{\varepsilon}\wedge$$
$$\langle\underline{\rho},\langle[\![\mathsf{S}]\!],\rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L}' : \mathsf{B}']\!]\}$$
$$\cup\{\langle\langle at[\![\mathsf{S}]\!],\rho\rangle\langle aft[\![\mathsf{S}]\!],\rho\rangle,\mathsf{R}''\rangle|\mathcal{B}[\![\mathsf{B}]\!]\rho = F\wedge \mathsf{R}' \notin \mathbb{R}_{\varepsilon}\wedge$$
$$\langle\underline{\rho},\langle at[\![\mathsf{S}]\!],\rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L}' : \mathsf{B}']\!] \wedge \langle\mathsf{L}'' : \mathsf{B}'',\mathsf{R}''\rangle = \mathsf{fstnxt}(\mathsf{R}')\wedge$$
$$\langle\underline{\rho},\langle aft[\![\mathsf{S}]\!],\rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L}'' : \mathsf{B}'']\!]\}$$
$$\text{while } \mathsf{fstnxt}(\mathsf{R}) = \langle\mathsf{L}' : \mathsf{B}',\mathsf{R}'\rangle$$

برای عبارت منظم $\mathsf{R} \in \mathbb{R}^{\dagger} \cup \mathbb{R}^{+}$ و $\mathsf{S} = \ \mathsf{if}\ (\mathsf{B})\ \mathsf{S_t}\ \mathsf{else}\ \mathsf{S_f}$ داریم:

$$\blacktriangleleft\ \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}\rangle[\![\mathsf{S}]\!] =$$
$$\{\langle\langle at[\![\mathsf{S}]\!],\rho\rangle,\mathsf{R}'\rangle|\langle\underline{\rho},\langle at[\![\mathsf{S}]\!],\rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L}' : \mathsf{B}']\!]\}$$
$$\cup\{\langle\langle at[\![\mathsf{S}]\!],\rho\rangle\langle at[\![\mathsf{S_t}]\!],\rho\rangle\pi,\mathsf{R}''|\mathcal{B}[\![\mathsf{B}]\!]\rho = T\wedge$$
$$\langle\underline{\rho},\langle at[\![\mathsf{S}]\!],\rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L}' : \mathsf{B}']\!]\wedge$$
$$\langle\langle at[\![\mathsf{S_t}]\!],\rho\rangle\pi,\mathsf{R}''\rangle \in \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}'\rangle[\![\mathsf{S_t}]\!]\}$$
$$\cup\{\langle\langle at[\![\mathsf{S}]\!],\rho\rangle\langle at[\![\mathsf{S_f}]\!],\rho\rangle\pi,\mathsf{R}''|\mathcal{B}[\![\mathsf{B}]\!]\rho = F\wedge$$
$$\langle\underline{\rho},\langle at[\![\mathsf{S}]\!],\rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L}' : \mathsf{B}']\!]\wedge$$
$$\langle\langle at[\![\mathsf{S_f}]\!],\rho\rangle\pi,\mathsf{R}''\rangle \in \hat{\mathcal{M}}^{\dagger}\langle\underline{\rho},\mathsf{R}'\rangle[\![\mathsf{S_f}]\!]\}$$

با توجه به موارد قبلی می‌شود مفهوم این دو مورد از تعریف را هم فهمید و پیچیدگی بیشتری ندارد. در باره‌ی مورد اول داستان به این شکل است که مانند تعریف‌های قبلی ابتدا تطابق لیترال اول عبارت منظم با ردهای پیشوندی تک عضوی با برچسب شروع دستور و محیط دلخواه بررسی می‌شود، سپس به این‌ها ردهای پیشوندی‌ای که در درون دستور $\mathsf{S_t}$ هستند و در موقعیت ابتداییشان عبارت بولی معنای صحیح دارد و البته با اولین لیترال باقی عبارت منظم هستند هم در زوج‌هایی

که از یک دانه از ردهای پیشوندی و یک عبارت منظم که بخش تطبیق داده نشده‌ی R با رد پیشوندی تشکیل شده‌اند، به مجموعه‌ی خروجی اضافه می‌شوند. دو مجموعه‌ی دیگری که در اینجا با خروجی‌ای که تاحالا توصیف کرده‌ایم اجتماع گرفته شده‌اند هم مربوط به حالتی است که در ابتدای رد پیشوندی عبارت بولی معنای غلط دارد و در این صورت چون برخلاف حالت قبل که عبارت منظم تطبیق داده نشده با خود تابع $\hat{\mathcal{M}}^{\dagger}$ مشخص می‌شد، در این حالت به ازای اینکه این ادامه‌ی عبارت منظم تهی است یا خیر دو حالت را برای تعریف آن داریم که در واقع عضو دوم زوج مرتب‌های موجود در خروجی را مشخص می‌کنند.

برای عبارت منظم $\mathsf{R} \in \mathbb{R}^{\dagger} \cup \mathbb{R}^{+}$ و $\mathsf{S} = \ \mathsf{break;}$ داریم:

$$\blacktriangleleft\ \hat{\mathcal{M}}^{\dagger}\langle \underline{\rho}, \mathsf{R}\rangle[\![\mathsf{S}]\!] =$$

$$\{\langle\langle at[\![\mathsf{S}]\!], \rho\rangle, \mathsf{R}'\rangle | \langle \underline{\rho}, \langle at[\![\mathsf{S}]\!], \rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L} : \mathsf{B}]\!]\}$$
$$\cup\{\langle\langle at[\![\mathsf{S}]\!], \rho\rangle\langle brk-to[\![\mathsf{S}]\!], \rho\rangle, \varepsilon\rangle | \mathsf{R}' \in \mathbb{R}_{\varepsilon}\wedge$$
$$\langle \underline{\rho}, \langle at[\![\mathsf{S}]\!], \rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L} : \mathsf{B}]\!]\}$$
$$\cup\{\langle\langle at[\![\mathsf{S}]\!], \rho\rangle\langle brk-to[\![\mathsf{S}]\!], \rho\rangle, \mathsf{R}''\rangle | \mathsf{R}' \notin \mathbb{R}_{\varepsilon}\wedge$$
$$\langle \underline{\rho}, \langle at[\![\mathsf{S}]\!], \rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L} : \mathsf{B}]\!] \wedge \langle \mathsf{L}' : \mathsf{B}', \mathsf{R}''\rangle = \mathsf{fstnxt}(\mathsf{R}')\wedge$$
$$\langle \underline{\rho}, \langle brk-to[\![\mathsf{S}]\!], \rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L}' : \mathsf{B}']\!]\}$$

با توجه به موارد قبلی این که این قسمت از تعریف چه معنایی دارد و به چه علت به این شکل است، قابل درک است.

برای عبارت منظم $\mathsf{R} \in \mathbb{R}^{\dagger} \cup \mathbb{R}^{+}$ و $\mathsf{S} = \ \mathsf{while\ (B)\ S_b}$: داریم:

$$\blacktriangleleft\ \hat{\mathcal{M}}^{\dagger}\langle \underline{\rho}, \mathsf{R}\rangle[\![\mathsf{S}]\!] = lfp^{\subseteq}\ (\hat{\mathcal{F}}^{\dagger}\langle \underline{\rho}, \mathsf{R}\rangle[\![\mathsf{S}]\!])$$

$$\mathsf{while}\ \hat{\mathcal{F}}^{\dagger}\langle \underline{\rho}, \mathsf{R}\rangle X = \{\langle\langle at[\![\mathsf{S}]\!], \rho\rangle, \mathsf{R}'\rangle | \rho \in \mathbb{EV} \wedge \langle \underline{\rho}, \langle at[\![\mathsf{S}]\!], \rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L}' : \mathsf{B}']\!]\}$$
$$\cup\{\langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle\langle aft[\![\mathsf{S}]\!], \rho\rangle, \varepsilon\rangle | \langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle, \varepsilon\rangle \in X\wedge$$
$$\mathcal{B}[\![\mathsf{B}]\!]\rho = F\}$$
$$\cup\{\langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle\langle aft[\![\mathsf{S}]\!], \rho\rangle, \varepsilon\rangle | \langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle, \mathsf{R}''\rangle \in X\wedge$$
$$\mathcal{B}[\![\mathsf{B}]\!]\rho = F \wedge \mathsf{R}'' \notin \mathbb{R}_{\varepsilon} \wedge \langle \mathsf{L}' : \mathsf{B}', \mathsf{R}'\rangle = \mathsf{fstnxt}(\mathsf{R}'') \wedge \mathsf{R}' \in \mathbb{R}_{\varepsilon}\wedge$$
$$\langle \underline{\rho}, \langle at[\![\mathsf{S}]\!], \rho\rangle\rangle \in \mathcal{S}^{r}[\![\mathsf{L}' : \mathsf{B}']\!]\}$$
$$\cup\{\langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle\langle aft[\![\mathsf{S}]\!], \rho\rangle, \mathsf{R}'\rangle | \langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle, \mathsf{R}''\rangle \in X\wedge$$
$$\mathcal{B}[\![\mathsf{B}]\!]\rho = F \wedge \mathsf{R}'' \notin \mathbb{R}_{\varepsilon} \wedge \langle \mathsf{L}' : \mathsf{B}', \mathsf{R}'\rangle = \mathsf{fstnxt}(\mathsf{R}'') \wedge \mathsf{R}''' \notin \mathbb{R}_{\varepsilon}\wedge$$

$$\langle \underline{\rho}, \langle at[\![\mathsf{S}]\!], \rho\rangle\rangle \in \mathcal{S}^r[\![\mathsf{L}' : \mathsf{B}']\!] \wedge \langle \mathsf{L}'' : \mathsf{B}'', \mathsf{R}'\rangle = \mathsf{fstnxt}(\mathsf{R}''')\wedge$$
$$\langle \underline{\rho}, \langle aft[\![\mathsf{S}]\!], \rho\rangle\rangle \in \mathsf{S}^r[\![\mathsf{L}'' : \mathsf{B}'']\!]\}$$
$$\cup\{\langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle\langle at[\![\mathsf{S_b}]\!], \rho\rangle\pi_3, \varepsilon\rangle | \langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle, \varepsilon\rangle \in X\wedge$$
$$\mathcal{B}[\![\mathsf{B}]\!]\rho = T \wedge \langle at[\![\mathsf{S_b}]\!], \rho\rangle\pi_3 \in \mathcal{S}^*[\![\mathsf{S_b}]\!]\}$$
$$\cup\{\langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle\langle at[\![\mathsf{S_b}]\!], \rho\rangle\pi_3, \varepsilon\rangle | \langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle, \mathsf{R}''\rangle \in X\wedge$$
$$\mathcal{B}[\![\mathsf{B}]\!]\rho = T \wedge \mathsf{R}'' \notin \mathbb{R}_\varepsilon \wedge \langle \mathsf{L} : \mathsf{B}, \varepsilon\rangle = \mathsf{fstnxt}(\mathsf{R}'')\wedge$$
$$\langle \underline{\rho}, \langle at[\![\mathsf{S}]\!], \rho\rangle\rangle \in \mathcal{S}^r[\![\mathsf{L} : \mathsf{B}]\!] \wedge \langle at[\![\mathsf{S_b}]\!], \rho\rangle\pi_3 \in \mathcal{S}^*[\![\mathsf{S_b}]\!]\}$$
$$\cup\{\langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle\langle at[\![\mathsf{S_b}]\!], \rho\rangle\pi_3, \mathsf{R}'\rangle | \langle \pi_2\langle at[\![\mathsf{S}]\!], \rho\rangle, \mathsf{R}''\rangle \in X\wedge$$
$$\mathcal{B}[\![\mathsf{B}]\!]\rho = T \wedge \mathsf{R}'' \notin \mathbb{R}_\varepsilon \wedge \langle \mathsf{L} : \mathsf{B}, \mathsf{R}''''\rangle = \mathsf{fstnxt}(\mathsf{R}'')\wedge$$
$$\langle \underline{\rho}, \langle at[\![\mathsf{S}]\!], \rho\rangle\rangle \in \mathcal{S}^r[\![\mathsf{L} : \mathsf{B}]\!] \wedge \mathsf{R}'''' \notin \mathbb{R}_\varepsilon\wedge$$
$$\langle \mathsf{L}' : \mathsf{B}', \mathsf{R}'''\rangle = \mathsf{fstnxt}(\mathsf{R}'''') \wedge \langle \underline{\rho}, \langle at[\![\mathsf{S_b}]\!], \rho\rangle\rangle \in \mathcal{S}^r[\![\mathsf{L}' : \mathsf{B}']\!]\wedge$$
$$\langle\langle at[\![\mathsf{S_b}]\!], \rho\rangle\pi_3, \mathsf{R}'\rangle \in \hat{\mathcal{M}}^\dagger\langle \underline{\rho}, \mathsf{R}'''\rangle[\![\mathsf{S_b}]\!]\}$$
$$\text{and } \langle \mathsf{L}' : \mathsf{B}', \mathsf{R}'\rangle = \mathsf{fstnxt}(\mathsf{R})$$

برای دستور تفاوتی که با سایر دستورات وجود دارد، حضور یک تابع در تعریف آن است و در واقع وارسی مدل به صورت کوچک‌ترین نقطه ثابت این تابع تعریف می‌شود. این همان کاری است که در تعریف معنای اجزای زبان هم انجام شد و وقتی که می‌خواهیم ساختار زبان را به صورت وارسی مدل اضافه انتظار می‌داشتیم که سر و کله‌ی عملگر نقطه ثابت هم پیدا شود، همان طور که تعریف بقیه‌ی دستورات زبان مطابق تعریف معانیشان که در ابتدای کار آورده‌ایم، انجام شده.

برای عبارت منظم $\mathsf{R} \in \mathbb{R}^\dagger \cup \mathbb{R}^+$ و $\mathsf{S} = \ ;$: داریم:

$$\blacktriangleleft\ \hat{\mathcal{M}}^\dagger\langle \underline{\rho}, \mathsf{R}\rangle[\![\mathsf{S}]\!] = \{\langle\langle at[\![\mathsf{S}]\!], \rho\rangle, \mathsf{R}'\rangle | \langle \underline{\rho}, \langle at[\![\mathsf{S}]\!], \rho\rangle\rangle \in \mathcal{S}^r[\![\mathsf{L} : \mathsf{B}]\!]\}$$
$$\text{while } \mathsf{fstnxt}(\mathsf{R}) = \langle \mathsf{L} : \mathsf{B}, \mathsf{R}'\rangle$$

برای عبارت منظم $\mathsf{R} \in \mathbb{R}^\dagger \cup \mathbb{R}^+$ و $\mathsf{S} = \{\mathsf{Sl}\}$: داریم:

$$\blacktriangleleft\ \hat{\mathcal{M}}^\dagger\langle \underline{\rho}, \mathsf{R}\rangle[\![\{\mathsf{Sl}\}]\!] = \hat{\mathcal{M}}^\dagger\langle \underline{\rho}, \mathsf{R}\rangle[\![\mathsf{Sl}]\!]$$

همین طور صادق بودن یک خاصیت $\mathsf{R} \in \mathbb{R}$ را برای برنامه‌ی P و محیط اولیه‌ی $\underline{\rho}$ با

$$\mathsf{P}, \underline{\rho} \models_s \mathsf{R}$$

نشان می‌دهیم و برقرار بودن این شرط به شکل زیر تعریف می‌شود:

$$\mathsf{P}, \underline{\rho} \models_s \mathsf{R} \iff \{\underline{\rho}\} \times \mathcal{S}^*[\![\mathsf{P}]\!] \subseteq \hat{\mathcal{M}}\langle \underline{\rho}, \mathsf{R}\rangle\mathcal{S}^*[\![\mathsf{P}]\!]$$

در اینجا تعریف وارسی مدل ساختارمند به پایان می‌رسد.

فصل ۶

ایمنی و سر زندگی

دو دسته‌ی معروف از خاصیت‌های مورد بررسی درباره‌ی یک برنامه وجود دارند که نام آن دو ایمنی و سرزندگی است. در این فصل ابتدا به تعریف این دو خاصیت می‌پردازیم و سپس در مورد اینکه آیا می‌شود این خواص را با سیستمی که داریم بررسی کنیم یا نه بحث می‌کنیم.

۱.۶ درستی و تمامیت

در این بخش سعی شده درستی و تمامیت این روش نسبت به خواص معرفی شده در این فصل، بر اساس تعریفی از درستی و تمامیت که در [۱۴] آمده بررسی شود. محتویات این بخش نیز در [۶] نیامده.

فصل ۷
نتیجه گیری

واژه‌نامه فارسی به انگلیسی

واژه‌نامه انگلیسی به فارسی

Bibliography

[1] Committee to review chinook zd 576 crash. report from the select committee on chinook zd 576., Feb 2002.

[2] A. S. E. Al. Mars climate orbiter mishap investigation boord phase i report., November 1999.

[3] A. Chlipala. *Certified Programming with Dependent Types: A Pragmatic Introduction to Coq Proof Assistant.* MIT Press, 2022.

[4] E. M. Clarke and E. A. Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In D. Kozen, editor, *Logic of Programs*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71. Springer, 1981.

[5] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model checking.* MIT Press, London, Cambridge, 1999.

[6] P. Cousot. Calculational design of a regular model checker by abstract interpretation. In R. M. Hierons and M. Mosbah, editors, *ICTAC*, volume 11884 of *Lecture Notes in Computer Science*, pages 3–21. Springer, 2019.

[7] P. Cousot. *Principals of Abstract Interpretation.* MIT Press, 2021.

[8] P. Cousot and R. Cousot. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *POPL '77: Proceedings of the 4th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 238–252. ACM Press, 1977.

[9] M. Davis and E. Weyuker. *Computability, Complexity, and Languages.* Academic Press, New York, 1983.

[10] D. Harel, D. Kozen, and J. Tiuryn. Dynamic logic. In *Handbook of philosophical logic*, pages 99–217. Springer, 2001.

[11] C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, 1969.

[12] M. Huth and M. Ryan. *Logic in computer science : modelling and reasoning about systems.* Cambridge University Press, Cambridge [U.K.]; New York, 2004.

[13] R. M. John E. Hopcroft and J. D. Ullman. *Introduction to automata theory, languages, and computation.* Addison-Wesley, 2003.

[14] X. R. K. Yi. *Introduction to Static Analysis: An Abstract Interpretation Perspective.* MIT Press, 2020.

[15] S. Kleene. Representation of Events in Nerve Nets and Finite Automata. In C. Shannon and J. McCarthy, editors, *Automata Studies*, pages 3–41. Princeton University Press, 1956.

[16] D. Koze. On kleene algebras and closed semirings. *Springer Berlin Heidelberg*, 1990.

[17] S. A. Kripke. A completeness theorem in modal logic1. *The journal of symbolic logic*, 24(1):1–14, 1959.

[18] J. Lions. Ariane 5 Flight 501 Failure: Report of the Inquiry Board, July 1996.

[19] M. Mukund. Linear-time temporal logic and buchi automata. *Tutorial talk, Winter School on Logic and Computer Science, Indian Statistical Institute, Calcutta*, 1997.

[20] G. J. Myers, C. Sandler, and T. Badgett. *The art of software testing.* John Wiley & Sons, Hoboken and N.J, 3rd ed edition, 2012.

[21] B. C. Pierce, A. Azevedo de Amorimand Chris Casinghino, M. Gaboardi, M. Greenberg, C. Hriţcu, V. Sjöberg, A. Tolmach, and B. Yorgey. *Programming Language Foundations.* Software Foundations series, volume 2. Electronic textbook, May 2018.

[22] H. G. Rice. Classes of recursively enumerable sets and their decision problems. *Transactions of the American Mathematical Society*, 74(2):358–366, 1953.

[23] A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific journal of Mathematics*, 5(2):285–309, 1955.

[24] G. Winskel. *The formal semantics of programming languages - an introduction.* Foundation of computing series. MIT Press, 1993.

[25] P. Wolper. Temporal logic can be more expressive. *Inf. Control.*, 56(1/2):72–99, January/February 1983.

Abstract

Abstract goes here...

College of Science
School of Mathematics, Statistics, and Computer Science

Thesis Title

Author name

Supervisor: name
Co-Supervisor: name
Advisor: name

A thesis submitted to Graduate Studies Office
in partial fulfillment of the requirements for the degree of
B.Sc./Master of Science/Doctor of Philosophy in
Pure Mathematics/ Applied Mathematics/ Statistics/ Computer Science

yyyy