
Západočeská univerzita v Plzni
Katedra informatiky a výpočetní techniky
Univerzitní 8
306 14 Plzeň

# Využití distribuované umělé inteligence ve strategických informačních systémech

Odborná práce ke státní doktorské zkoušce

Petr Pinkas

Technická zpráva č. DCSE/TR-2014-3
Srpen, 2014

Distribuce: veřejná

Technická zpráva č. DCSE/TR-2014-3
Srpen, 2014

# Využití distribuované umělé inteligence ve strategických informačních systémech

Petr Pinkas

## Abstrakt

Z hlediska koncepce návrhu (nejen informačních) systémů je z oboru distribuované umělé inteligence zajímavá právě oblast multiagentních systémů. Výzkumu v této oblasti se dosud věnovala nemalá pozornost, přesto v praxi (pokud vynecháme uplatnění v simulacích) lze stále spatřit nanejvýše náznaky multiagentního paradigmatu. Je již zřejmé, že při návrhu spolehlivých multiagentních systémů nepostačují obvyklé (intuitivní a empirické) přístupy bez dostatečné opory v hlubší formalizaci.

Zmíněnou oporu se přirozeně nabízí hledat v odvětví formálních metod, zejména pak v již poměrně rozvinuté oblasti procesních kalkulů (či algeber). Předkládaná práce se po úvodní úvaze soustředí na možnost využití Milnerovo kalkulu (CCS), který lze chápat jako rámec pro rodinu operačně sémantických modelů založených na přechodových systémech. Avšak i zde narazíme na překážku v podobě dosud neuspokojivě vyřešeného problému ekvivalence přechodových systémů. Formálních (a zdánlivě rozumných) ekvivalencí lze totiž najít celou řadu a problém tak spočívá spíše v korespondenci mezi formální a skutečně pozorovatelnou (empirickou) ekvivalencí. Přitom právě odpovídající ekvivalence hraje při hlubší formalizaci klíčovou roli – vždyť pojem ekvivalence úzce souvisí s pojmem chování (dokonce lze jeden pojem definovat druhým) a chování je předmětem návrhu (nejen multiagentních) systémů.

Proto jsou v předkládané práci zevrubně prozkoumány přechodové systémy i způsob vybudování Milnerovo kalkulu. Na základě poznatků je navržen přístup vedoucí k odpovídající formalizaci chování (a tedy i ekvivalence) a následně k algebraické struktuře patřičně upřesňující Milnerovo kalkulus. Závěrem je naznačen i směr další práce.

Kopie zprávy jsou dostupné na
http:/www.kiv.zcu.cz/cz/vyzkum/publikace/technicke-zpravy/
nebo na žádost poslanou na následující adresu:

Západočeská univerzita v Plzni
Katedra informatiky a výpočetní techniky
Univerzitní 8
306 14 Plzeň
Česká republika

# Poděkování

Tímto bych rád poděkoval své školitelce Doc. Dr. Ing. Janě Klečkové ze Západočeské univerzity v Plzni za její vedení v mém doktorském studiu a především za její trpělivost. Také bych rád poděkoval svým přátelům a blízkým za jejich podporu.

# Obsah

# 1 Úvod

**Multiagentní systémy**

V popředí zájmu distribuované umělé inteligence jsou *multiagentní systémy*, které jsou tvořeny vždy skupinou volně vázaných autonomních systémů – *agentů*. Přitom výsledná funkce multiagentního systému je dána dynamikou vzájemných interakcí agentů (podle komplexnosti interakcí potom lze hovořit o koordinaci či dokonce kooperaci). Zřejmou inspirací zde podle Štěpánkové a spol. byla skutečnost, že „sdružování jednotlivců do větších skupin je jednou z charakteristických vlastností živé hmoty; schopnosti vzniklé skupiny výrazně převyšují prostý součet všeho, co je schopen dosáhnout každý z jejích členů“ [11].

Paradigma multiagentních systémů nachází uplatnění hlavně v robotice, potažmo kybernetice. Avšak lze se domnívat, že toto paradigma by mohlo najít uplatnění také v softwarovém inženýrství. Vždyť informační (a obecně softwarové) systémy se potýkají s podobnými dvěma problémy, které se pomocí multiagentního paradigmatu snaží řešit i kybernetika.

Prvním problémem je (trvale rostoucí) komplexnost systémů. Například Mařík a spol. soudí, že „velký zájem o multiagentní paradigma je zcela přirozený, protože řada systémů, včetně systémů softwarových, dosáhla takového stupně složitosti, že je není možné řídit a provozovat jako systémy monolitické“ [12]. Přičemž dodejme, že takové systémy není obtížné jen provozovat, ale dokonce i navrhnout jako monolitické. Ferber navíc upřesňuje, že „složitost problému si vynucuje lokální úhel pohledu; jakmile jsou problémy příliš rozsáhlé na to, aby mohly být analyzovány jako celek, tak řešení založená na lokálních přístupech často umožňují vyřešení takových problémů mnohem snáze“ [13]. Softwarové inženýrství samozřejmě má metody pro dekompozice systémů, avšak ve své podstatě (nezávisle na použitém přístupu – procedurálním, objektovém či jiném) se stále jedná o dekompozice algoritmické. Jinými slovy, zmiňovaná monolitičnost návrhu spočívá ve snaze obsáhnout funkci celého systému v důsledku jedním (byť i paralelizovaným) algoritmem. Přitom komponenty systému (dané algoritmickou dekompozicí) jsou pevně vázané a nepředstavují nic jiného, než výseky globálního úhlu pohledu.

Druhým problémem je (přirozená) distribuovanost systémů doprovázená volnějšími vazbami jejich komponent. V tomto ohledu Ferber připomíná, že už jen „počítačové sítě nás nutí přijmout distribuovaný pohled“ a ihned (s trochou nadsázky) prohlašuje, že „ve věku meziplanetárních sítí, kdy všechna data i výpočetní výkon jsou distribuována do značného počtu lokalit, softwaroví inženýři musí přestat ‚myslet globálně‘“ [13]. Následně však Ferber zcela pragmaticky (a velmi trefně) vystihuje: „softwarové inženýrství se podílí na vytváření autonomních modulů, které mohou vzájemně interagovat dokonce i tehdy, a zvláště tehdy, když byly navrženy různými lidmi, různými týmy nebo různými společnostmi“ [13]. Právě (přirozená) distribuovanost s volnějšími vazbami může vést k druhému extrému, který bychom mohli nazvat naivně lokálním přístupem (a který dnes lze často spatřit v oblasti webových služeb). Tento přístup spočívá v představě, že pokud každý autonomní modul bude správně fungovat (tzn. plnit, z lokálního úhlu pohledu, svůj účel) a bude správně interagovat s bez-

prostředními moduly (tzn. dodržovat příslušná rozhraní, protokoly apod.), tak bude funkce vzniklého systému také správná. Tímto úskalím se v širších souvislostech zabývají například Smith a Phillips, a upozorňují, že „na aktuálně vyvíjených systémech vidíme další jev přispívající ke znepokojení: absenci celkové zodpovědnosti za systém systémů“ [14].

Z právě uvedeného popisu obou problémů vyplývá, že v určitých případech bychom potřebovali nějaký „vyvážený“ přístup nacházející se někde mezi monoliticky globálním a naivně lokálním přístupem. Jinými slovy, v určitých případech sice nevyžadujeme či nedokážeme veškeré děje v systému plně (algoritmicky) předepsat, ale na druhou stranu požadujeme, aby nežádoucí jevy (nikdy) nenastaly a přitom potřebné jevy (nakonec) nastaly. Dovolme si zde pro ilustraci zjednodušený příměr s mravenci (tzn. volně vázanými autonomními komponenty systému) a stavbou mraveniště (tzn. funkcí systému). Nezáleží zda každý mravenec břemeno (z globálního úhlu pohledu) nese po optimální trajektorii, případně ho několikrát upustí či vymění (z lokálního úhlu pohledu vykonává svou funkci vždy správně). Důležité však je, že souběh všech manipulací mravenců s břemeny (nakonec) vede ke stavbě mraveniště. A právě paradigma multiagentních systémů by nastíněný „vyvážený“ přístup mohlo přinést. Ferber se dokonce domnívá, že „multiagentní systémy zde hrají důležitou roli možného nástupce objektově orientovaných systémů“ [13].

Proč tedy v softwarovém inženýrství dosud není multiagentní paradigma široce uplatňováno? Multiagentní paradigma sice odpovídá na oba výše popsané problémy, avšak zároveň přináší jiné obtíže. Funkce multiagentního systému je totiž dána dynamikou (tzn. možnými souběhy) interakcí mezi agenty. Přitom ověřit správnost této dynamiky není možné testováním (které je v softwarovém inženýrství široce využívané), takže už jen z tohoto důvodu se využití formálních metod stává nutností (a nejen v případě systémů strategických). Avšak formální metody dosud bohužel nedosáhly dostatečného pokroku. Potom se nelze divit, že softwarové multiagentní systémy (pokud vynecháme uplatnění v simulacích) mají podobu spíše klasických distribuovaných systémů (typickým příkladem je systém, který prezentují autoři Martí a spol. [15]). Vždyť bez dostatečné opory ve formálních metodách je taková opatrnost pochopitelná a zcela na místě.

Obdobně situaci vidí i Mařík a spol., když uvádí, že „většina dnes existujících multiagentních aplikací jsou systémy vyvinuté ad hoc, přičemž využívají jenom těch nejjednodušších, obvykle pouze reaktivních, modelů chování; architektury agentů i globální architektury multiagentních systémů jsou navrhovány spíše intuitivně, bez využití hlubší formalizace, která by umožňovala plánovat a korektně realizovat i složitější scénáře vyžadující přesně koordinovanou součinnost většího počtu agentů“ [12]. Přitom za zmíněnou hlubší formalizaci jistě nelze považovat pouhý formální zápis (tzn. popis v návrhových nebo programovacích jazycích). Jak upozorňuje Štěpánková a spol., tak „metodika návrhu multiagentního řešení musí vycházet z důkladného porozumění problematice součinnosti agentů“ [11]. A porozumění problematice součinnosti (resp. dynamiky interakcí) agentů je možné až teprve prostřednictvím formálního uvažování. Tím se opět dostáváme k formálním metodám.

**Formální metody**

Formální metody jsou odvětvím počítačových věd, které se zabývá výzkumem principů a vývojem technik pro rigorózní (nejlépe zcela formální) modelování a analyzování softwarových i hardwarových systémů. Nezbytným rysem je právě rigorózní přístup, který je běžný

v matematice a který brání nejednoznačnosti či nekonzistenci v úvahách i příslušných úkonech. Zcela formální přístup navíc umožňuje mechanické (strojové) provádění úkonů, díky čemuž lze (oproti manuálnímu provádění) eliminovat lidské omyly a zvládnout rozsáhlejší úlohy. Nicméně základní motivací k rozvoji odvětví formálních metod je předpoklad, že přinese do softwarového (i hardwarového) inženýrství hlubší porozumění navrhovaným systémům a větší jistotu při jejich návrhu, obdobně jako přinesla aplikovaná matematika do ostatních inženýrských oborů. Baier a Katoen dokonce uvádí, že „řečeno v kostce, formální metody mohou být považovány za ‚aplikovanou matematiku pro modelování a analyzování ICT systémů'" [16]. Z užšího hlediska jsou pak formální metody technikami pro specifikaci a verifikaci zvolených vlastností softwarových i hardwarových systémů.

Pohlížení na odvětví formálních metod a na dosažené výsledky se však postupem času stalo značně kontroverzním. Část odborné veřejnosti je obhajuje, když například Baier a Katoen připomínají, že „během posledních dvou desetiletí vedl výzkum v rámci odvětví formálních metod k vývoji některých velmi slibných verifikačních technik, které usnadňují včasné rozpoznání chyb; tyto techniky jsou provázeny výkonnými softwarovými nástroji, které mohou být použity k automatizaci řady verifikačních kroků; a výzkumy ukázaly, že formální verifikační postupy by odhalily chyby, které se projevily například u rakety Ariane-5, sondy Mars Pathfinder, procesoru Intel Pentium II a přístroje na radiační terapii Therac-25" [16]. Avšak zbývající část odborné veřejnosti je přinejmenším skeptická. Kritikou nešetří například Parnas, jenž se také ohlíží a připomíná, že „je to již více než 40 let, co nám zesnulý Robert Floyd ukázal, jak ‚programům přiřadit význam' a předvedl, jak bychom mohli ověřit, že programy budou dělat to, co dělat mají" a přitom „aplikace formálních metod v běžné praxi zůstávají takovými výjimkami, že tím potvrzují, že využívání formálních metod není běžnou záležitostí" [17]. Parnas následně vysvětluje, že „někdy si autoři pouze hrají se slovíčky; například technika vkládání odlaďovacích příkazů do kódu, kterou mě učili v roce 1959, byla nedávno vyhlašována jako ‚užití *assertions* v běžné praxi'; v ostatních případech bližší zkoumání odhalí opravdu heroické úsilí s velmi složitými formálními modely, avšak jen pramálo důkazů o správnosti skutečného kódu; tyto snahy často nevedou k opakovanému užití nebo širšímu přijetí takové metody; vývojářské organizace, které běžně používají takové metody pro skutečné produkty, jsou vzácné" [17]. V závěru pak Parnas vyzývá: „musíme se naučit používat matematiku v oblasti vývoje softwaru, ale musíme zpochybnit, a být připraveni zavrhnout, většinu metod, které jsme diskutovali a prosazovali po všechny ty roky; musíme přezkoumat předpoklady, na kterých jsou tyto metody založeny, a zjistit, které z nich obstojí" [17].

Příznačně kontroverzní je pak i reakce na Parnasovu kritiku. Plaks s kritikou souhlasí, když píše: „mnoho lidí pracujících v oblasti počítačových věd, ale v inženýrském prostředí (třeba já jsem elektroinženýr se zaměřením na výpočetní techniku), již dlouho přemýšlelo, debatovalo a psalo články o těch stejných problémech, o kterých Parnas pojednává; a jak Parnas upozorňuje, tak lidé pracující v reálném světě nepoužívají formální metody, protože neposkytují řešení, které hledáme" [18]. Zatímco Pike reaguje řadou výtek, na které pak Parnas odpovídá (mimo jiné) takto: „je pravdou, že jsem opomněl zmínit ‚pokročilejší přístupy jako *model checkers* a *satisfiability modulo theories solvers*'; existují stovky zajímavých technik, které jsem nezmínil; jsou to cenné výsledky výzkumu, avšak neutvářejí metodu pro vývoj softwaru, která je připravena pro široké použití" a takto: „je pravdou, že mnoho firem financuje výzkum formálních metod; začaly s tím nejméně před 40 lety; a také došlo k omezenému úspěchu v některých snazších oblastech, jako je analýza protokolů, ale aplikace těchto metod

má daleko ke ‚standardní praktice'" [18]. Kritika však neznamená zavrhování celého odvětví formálních metod, spíše je vyjádřením nenaplněných očekávání. V souladu s tím také Plaks píše, že „přístup, v němž počítačoví vědci používají matematiku k výpočtu výsledků/řešení jako inženýři v mnoha jiných oblastech, si zaslouží více pozornosti a měl by být dále rozvíjen" [18]. Obdobně pak celou diskusi uzavírá i Parnas: „doufám, že jednoho dne budeme mít formální metody, které budou dostatečně účinné a praktické na to, aby je vývojáři softwaru běžně používali tak, jako inženýři v tradičních odvětvích používají diferenciální a integrální počet; dosud jsme k tomuto cíli moc nepokročili, a výzkumníci se musí ptát proč" [18].

Přetrvávající problémy se však netýkají jen praktické použitelnosti formálních metod, třebaže se tímto směrem kritika upírá nejčastěji. Mnoho problémů je hlubších, teoretických – jako například (zdánlivě elementární) ekvivalence chování dvou systémů. Formalizace softwarového inženýrství je zřejmě mnohem obtížnější úkol, než se předpokládalo. Situace tak silně připomíná zklamání, kterým si prošlo odvětví umělé inteligence poté, co na základě počátečních úspěchů vyvolalo příliš velká očekávání, která pak nedokázalo naplnit. Přitom však nelze nevyslovit nabízející se otázku. Proč je formální (ale i rigorózní) uvažování nad softwarovými systémy natolik obtížné, když samy softwarové systémy jsou ze své podstaty zcela formálními artefakty? Možnou odpověď lze spatřit v názoru Johnsona a Butlera: „problém se softwarem je, že složitost přesahuje naši schopnost mít nad ním intelektuální kontrolu; naše intuice a zkušenosti se pojí se spojitými systémy, avšak software vykazuje nespojité chování; jsme nuceni zvlášť zvažovat nebo testovat miliony posloupností diskrétních stavových přechodů" [19]. Velmi podobně smýšlí také Abramson a Pike: „na rozdíl od jiných inženýrských artefaktů, které jsou již matematicky modelované (namáhání mostu, aerodynamika letadla atd.), mnohé koncepty v počítačových vědách jsou úplně nové, a výzvou výzkumu prostě je, jak matematicky tyto koncepty modelovat" [20].

Asi největším úspěchem odvětví formálních metod (především z hlediska uplatnění v praxi) je vyvinutí techniky *model checking*. Osvědčila se již při verifikaci protokolů i hardwarových systémů, a snahou dalšího vývoje je její aplikace i na softwarové systémy. Tato technika je založena na vytvoření (konečného) modelu předmětného systému a na zkontrolování, zda přepsaná vlastnost je v takovém modelu splněna. Baier a Katoen názorněji vysvětlují: „*model checking* je verifikační technika, která (na vytvořeném modelu) tzv. hrubou silou prozkoumá všechny možné stavy systému; podobně jako počítačový šachový program prověřuje možné tahy, *model checker* (softwarový nástroj provádějící kontrolu modelu) systematicky vyzkouší všechny možné průběhy vývoje systému; tímto způsobem je možné prokázat, že daný model systému skutečně splňuje určenou vlastnost" [16]. Ověřované vlastnosti samozřejmě musí být popsány formálně, k čemuž se nejčastěji využívá některá temporální logika (např. LTL, CTL, HML). Velkou výhodou techniky *model checking* pak je, že umožňuje ověřovat poměrně rozmanité vlastnosti systému (resp. výroky o systému ve zmíněné logice), což techniku činí do značné míry všestrannou. Na druhou stranu má tato technika i řadu omezení. Z hlediska uplatnění v praxi je asi největším omezením, které Baier a Katoen uvádí, právě skutečnost, že technika „trpí problémem tzv. exploze stavového prostoru, tj. počet stavů nutných k přesnému namodelování systému může snadno překročit množství dostupné paměti počítače; a navzdory vývoji několika velmi účinných způsobů pro potlačení tohoto problému, modely reálných systémů mohou být stále příliš velké na to, aby se vešly do paměti" [16]. Z hlediska aplikace na multiagentní systémy však vyvstává jiné, principiální omezení. Model systému zde totiž slouží pouze k vytyčení stavového prostoru tak, aby mohl systematicky procházen

a mohlo být v každém stavu ověřeno splnění předepsané vlastnosti. Potom jednotlivé ověřované vlastnosti hrají roli pouze izolovaných a velmi konkrétních sond do systému (nelze tak usuzovat souvislosti, odvozovat předpovědi apod.). Stručně řečeno, technika *model checking* nám tedy neposkytuje aparát, pomocí kterého bychom mohli důkladně porozumět problematice součinnosti agentů (jak doporučuje Štěpánková a spol., viz výše).

Pro formální uvažování nad dynamikou (tzn. možnými souběhy) interakcí mezi agenty bychom potřebovali obdobný aparát, jakým je například aritmetika pro lineární algebru nebo diferenciální počet pro matematickou analýzu. Takovým aparátem by zřejmě mohl být nějaký *procesní kalkul* (či *procesní algebra*). Vždyť poměrně rozvinutá oblast procesních kalkulů (a algeber) se zabývá právě rozmanitými přístupy k formálnímu modelování souběžných systémů. Základní přístupy (z nichž vychází většina ostatních) však reprezentují dva původní kalkuly, tj. jednak *Communicating Sequential Processes* (CSP), jenž vyvinul Hoare [4], a jednak *Calculus of Communicating Systems* (CCS), který vybudoval Milner [8]. Podstatné rozdíly mezi oběma kalkuly přitom pramení z odlišných motivací. Hoarův vývoj CSP byl původně motivován výhradně imperativním programováním paralelních procesů komunikujících pomocí zpráv. V tomto ohledu je výmluvné Hoarovo ohlédnutí se zpět: „v rané verzi (CSP) se očekávalo, že všechny procesy v paralelním příkazu vždy skončí; důvodem byl předpoklad, že správnost procesu by mohla být specifikována stejně jako u konvenčního programu pomocí *postcondition*, tzn. pomocí predikátu, který má být splněn při úspěšném skončení“ [4]. Naproti tomu Milnerův vývoj CCS byl motivován snahou obecněji a plně postihnout pozorovatelné chování (tvořené synchronizovanou komunikací) souběžných systémů. Milner přitom zdůrazňuje, že „náš kalkulus (CCS) se odlišuje (od CSP a dalších přístupů směřujících k lepšímu porozumění paralelního programování) ve dvou ohledech: zaprvé, v žádném akceptovaném smyslu není imperativním jazykem – nemá žádné příkazy, jen výrazy; zadruhé, vyvinul se jako součást matematické studie“ [8]. Přestože se tedy postupem času oba kalkuly v mnoha ohledech sblížily, tak původní motivace v nich zanechávají nesmazatelné rozdíly. A nejdůležitějším takovým rozdílem je odlišné pojetí ekvivalence.

Hoare vysvětluje rozdílné pojetí ekvivalence následovně: „CCS zohledňuje mnoho odlišností mezi procesy, které by v této knize (CSP) byly považovány za totožné; důvodem je, že CCS má sloužit jako rámec pro rodinu modelů, z nichž každý smí vést k více ztotožněním než CCS, ale k méně nikoliv; a aby se neomezovala škála modelů, tak CSS sám vede jen k těm ztotožněním, které se zdají být naprosto nezbytné; nicméně v matematickém modelu v této knize (CSP) usilujeme přesně o opačný cíl – dosažení co nejvíce ztotožnění, zohledňujíce jen ty nejzásadnější odlišnosti“ [4]. Jinými slovy, existují i procesy, které ačkoliv jsou v Hoarovo CSP ekvivalentní, tak mají odlišné pozorovatelné chování (a tedy odlišný vliv na své okolí, se kterým komunikují, či abstraktněji vzato, jakkoliv interagují). Proč to však v Hoarovo přístupu nevadí? Protože CSP je zaměřen na vyšetřování jen některých jevů v komunikaci mezi procesy (jako je například *deadlock*), které jsou zajímavé především z hlediska paralelního programování (tedy z hlediska původní motivace vývoje CSP). Avšak v multiagentních (a jiných volně vázaných) systémech se díky volnějším vazbám připouští rozmanitější souběhy interakcí, čímž se mohou stát podstatnými i mnohem jemnější odlišnosti v pozorovatelném chování, než které zohledňuje CSP. Lépe řečeno, je naopak otázkou, jaké odlišnosti (a zda vůbec) lze pomíjet – tzn. jak jemná (či hrubá) by ekvivalence měla být. V tomto ohledu je zajímavé, jak Milnerův CCS charakterizuje právě Hoare: „cílem CCS bylo dosáhnout maximální vyjadřovací schopnosti s co nejmenším počtem elementárních operátorů; to je zdrojem

elegance a síly CCS, a výrazně zjednodušuje vyšetřování rodiny modelů definovaných různými ekvivalencemi“ [4].

Z dosud uvedeného tedy plyne, že při hledání aparátu vhodného pro formální uvažování nad dynamikou interakcí mezi agenty je vhodné zaměřit pozornost především na Milnerův přístup.

**Model, chování a ekvivalence**

Základem operačně sémantického modelu Milnerovo kalkulu (i ostatních procesních kalkulů) jsou *přechodové systémy*. A nezbytnou součástí tohoto modelu je pak také upřesnění, jaké odlišnosti mezi přechodovými systémy jsou pomíjeny a jaké již nikoliv. Různým upřesněním je tedy možné stanovit celou řadu (rodinu) modelů. Avšak proč se zabývat dalšími modely? Vhodným pomíjením odlišností vlastně upřesňujeme, jaký význam pro nás přechodové systémy mají. Přitom pomíjet se snažíme jen a právě takové odlišnosti, aby vyplývající význam odpovídal tomu, co chápeme pod pojmem chování přechodového systému (z hlediska určité aplikace, resp. třídy aplikací). Stručně vyjádřeno, různé (operačně sémantické) modely vystihují různá chápání pojmu chování. Přestože tedy můžeme říci, že prostřednictvím kalkulu budeme formálně uvažovat nad chováním přechodových systémů, musíme si uvědomit, že význam pojmu chování (a nad čím nakonec budeme vlastně uvažovat) podstatně závisí právě na modelu, resp. na zmiňovaném pomíjení (a zohledňování) odlišností.

Nyní je vhodné zdůraznit, že stanovení toho, jaké odlišnosti mezi přechodovými systémy jsou pomíjeny a jaké již nikoliv, není nic jiného, než stanovení ekvivalence přechodových systémů. Ačkoliv jsme se této skutečnosti dotkli již při diskusi o rozdílech mezi Milnerovo kalkulem (CCS) a Hoarovo kalkulem (CSP), tak je vhodné si důležitost ekvivalence plně uvědomit. Ekvivalence totiž neznamená jen zaměnitelnost. Vždyť vzhledem k předchozímu odstavci je to právě ekvivalence, která (třebaže implicitně) definuje pojem chování a která podstatně ovlivňuje, nad čím budeme prostřednictvím kalkulu formálně uvažovat.

Vezměme třeba *bisimulační ekvivalenci*, o níž Rabinovich říká, že „je uznávána jakožto nejjemnější ekvivalence chování, o které lze u souběžných systémů uvažovat; často je tvrzeno, že (silně) bisimulačně ekvivalentní přechodové systémy nejsou rozlišitelné žádným z rozumně pojatých pozorování“ [10]. Mimochodem, právě tuto ekvivalenci Milner [8, 1] nakonec využil pro stanovení (operačně sémantického) modelu svého kalkulu. Kdyby u ekvivalence šlo pouze o (zaručenou) zaměnitelnost, potom bychom si s bisimulační ekvivalencí jistě mohli vystačit. Z uvedené citace totiž plyne, že při jakémkoliv rozumném chápání pojmu chování jsou dva bisimulačně ekvivalentní přechodové systémy vzájemně zaměnitelné, aniž by došlo k jakékoliv změně v chování. Praktičtěji řečeno, takováto ekvivalence nám nabízí jistotu, zda například můžeme jednu implementaci nějakého agenta nahradit jinou implementací, nebo zda určitá implementace odpovídá dané specifikaci, apod. Sice při konkrétním a patřičném chápání pojmu chování, tzn. při patřičně hrubší ekvivalenci, bychom „uviděli“ více vhodných implementací, avšak to není nezbytně důvodem odmítnout bisimulační ekvivalenci a hledat či zkoumat ekvivalence další. Ovšem podstatně závažnější se situace stává, jakmile jde o formální (či alespoň rigorózní) uvažování. Vždyť kromě definic operací kalkulu je to právě ekvivalence, na které závisí vlastnosti (či axiomy) kalkulu a na které v důsledku závisí, co lze v rámci kalkulu odvodit nebo dokázat. Asi nejnázornější je to u důkazu sporem. Představme si, že nějaké tvrzení o chování dokážeme sporem. Přesněji

tím, že z opačného tvrzení a nějakého (pravdivého) předpokladu v rámci kalkulu odvodíme neplatnou rovnost. Avšak pokud je ekvivalence kalkulu jemnější, než jaká by odpovídala našemu skutečnému chápaní pojmu chování, tak dokázané tvrzení nemusí být ve skutečnosti pravdivé. V odpovídající (hrubší) ekvivalenci by totiž zmíněná rovnost již mohla platit, tím by ke sporu nedošlo a tvrzení by dokázáno nebylo. Je tedy zřejmé, jak důležité je najít ekvivalenci, která by (i svou jemností) odpovídala zamýšlenému pojetí chování. Proto bude v této práci brán zvláštní zřetel na pojetí ekvivalence a chování.

**Záměr práce**

V rámci této práce nejdříve podrobně prozkoumáme právě přechodové systémy (viz kapitola 2). Všímat si budeme především vlastností, jež by mohly být relevantní k chápání pojmů ekvivalence a chování. Nicméně z předchozích odstavců již víme, že tyto (úzce související) pojmy lze chápat různě. Milner pro upřesnění (resp. intuitivní vystižení záměru) hovoří o *pozorování* a vysvětluje, že „záměrem je popsat souběžný systém natolik, aby bylo určeno přesně, jaké chování uvidí či zakusí vnější pozorovatel; přístup je zcela extenzionální – dva systémy jsou nerozlišitelné, jestliže je nemůžeme shledat odlišnými bez prozkoumání jejich složení“ [8]. Pokud přijmeme, že pozorovatelné je i chování (implicitně) definované bisimulační ekvivalencí (tzn. nejjemnější rozumnou ekvivalencí), tak pojem pozorovatelné chování zůstává příliš široký. Proto jej dále upřesníme omezením, že vnější pozorovatel smí manipulovat s přechodovým systémem jen interakcemi prostřednictvím *akcí* (jimiž jsou podmíněny stavové přechody, viz oddíl 2.1). Takto upřesněné pozorovatelné chování pak označíme jako *vnější projev*. Právě takovýmto charakterem chování jednotlivých agentů je totiž tvořena dynamika interakcí v multiagentním systému.

Dále se seznámíme s Milnerovo kalkulem (viz kapitola 3), ovšem zajímat nás bude především způsob jakým jej Milner vybudoval. Kvůli hlubšímu porozumění však nebudeme konstrukci kalkulu z jeho knihy [1] pouze reprodukovat, ale provedeme stručnou rekonstrukci (tzn. zachováme původní myšlenkový postup, ale s vlastní formulací). Díky tomu si snadněji všimneme elegance i problémů, které Milnerův přístup přináší. Po zvážení následně navrhneme odlišný přístup (viz kapitola 4). Cílem je (explicitně) definovat vnější projev přechodových systémů, resp. najít jeho formálně uchopitelný popis. Tím bychom automaticky dostali i ekvivalenci přechodových systémů. Dalším cílem je nad množinou těchto popisů vybudovat algebraickou strukturu, která převezme jazyk i vlastnosti (či axiomy) Milnerovo kalkulu. Stručně řečeno, místo uplatnění nové ekvivalence (jež by definovala vnější projev) ve stávajícím postupu konstrukce Milnerovo kalkulu zvolíme jiný postup, který ovšem povede ke shodnému jazyku (a jeho shodné návaznosti na přechodové systémy). Hlavní ideje navrhovaného přístupu již byly i publikovány, zejména záměr formálně popsat důsledek akce v nedeterministickém přechodovém systému [21] a princip tzv. odkládaného nedeterminismu [22].

Nakonec se soustředíme na ekvivalenci (viz kapitola 5). Na Milnerovo výchozím pojetí ekvivalence (založené na tzv. derivačních stromech) si ukážeme, že nedostatečně formální pojetí může snadno vést i ke sporným tvrzením. Ovšem při formalizaci ekvivalence se zase snadno můžeme odchýlit od jejího skutečného chápání (např. už jen tím, že „sklouzneme“ k některé z forem, které na dané úrovni poznání dovedeme vyjádřit). Proto pro udržení návaznosti na onu skutečnost a pro ověření formálních pojetí ekvivalence použijeme *metaforu tlačítek*, která nám umožní provádět pomyslné experimenty (úkol metafory je podstatně snazší, má pouze

popsat zkušenost pomyslného experimentátora, takže riziko onoho „sklouznutí“ je mnohem menší). Následně se již podrobněji seznámíme s Milnerovo konečným pojetím ekvivalence v jeho kalkulu (tj. s již zmiňovanou bisimulační ekvivalencí) a provedeme porovnání s pojetím ekvivalence v navrhované algebraické struktuře (založené na nalezeném popisu vnějšího projevu přechodových systémů). Obě pojetí ekvivalence navíc posoudíme pomocí zmíněné metafory tlačítek.

# 2 Studie přechodového systému

## 2.1 Přechodový systém

Nejdříve se stručně seznámíme s přechodovým systémem samotným. Definovat jej budeme sice již s ohledem na potřeby modelování chování (resp. vnějšího projevu), avšak některé pojmy ponecháme obecnější (například pojem podmínka přechodu, která potom může být ztotožněna jak s akcí, tak se vstupním znakem). Tím zajistíme, aby definice již zachycovala vlastnosti podstatné pro modelování a přitom stále ještě umožňovala zamýšlené porovnání s automatem. Právě porovnáním s automatem si totiž snáze všimneme řady vlastností.

**Definice značeného přechodového systému**

Značený přechodový systém (dále jen přechodový systém) je abstraktním výpočetním modelem, který je založen na množině stavů a přechodech mezi nimi. Nejsou zde kladeny žádné požadavky na vlastnosti množin a jejich prvky jsou interpretovány jen minimálně. Pro naše potřeby si tedy přechodový systém upřesníme a interpretujeme. Jako upřesnění nám postačí, když množiny, na kterých je přechodový systém založen budou konečné. Potom je přechodový systém trojice

$$(S, \Lambda, R)$$

kde $S$ je konečná množina stavů, $\Lambda$ je konečná množina značek a $R$ je přechodová relace. Přesněji řečeno, $R \subseteq S \times \Lambda \times S$, přičemž platí

$$(A, \alpha, B) \in R \qquad\qquad A, B \in S \quad \alpha \in \Lambda$$

právě když existuje přechod ze stavu $A$ do stavu $B$ s označením $\alpha$. Vzhledem k tomu, že jsou přechody definovány uvedenou relací, tak je stav přechodového systému vždy jednoznačně určen (protože každý přechod vyústí do jednoznačně daného stavu).

Význam značek z množiny $\Lambda$ není definován. Podle potřeby mohou představovat příčinu, důsledek nebo jakoukoliv jinou vlastnost přechodu (např. pravděpodobnost přechodu). Pro naše potřeby budeme značky interpretovat jako příčinu přechodu, přesněji jako nutnou a nikoliv postačující podmínku přechodu. Ovšem s tím, že smí být splněna vždy nejvýše jedna z množiny podmínek $\Phi(A)$, kde $A$ je aktuálním stavem přechodového systému a $\Phi(X)$ definovaná jako

$$\Phi(X) = \{ \lambda : (X, \lambda, Y) \in R \}$$

představuje množinu všech podmínek, které přísluší některému z bezprostředních přechodů ze stavu $X$. Jinými slovy, jednak nepřipouštíme souběžné splnění dvou či více podmínek, ale také nepřipouštíme splnění takové podmínky, která nepřísluší některému z bezprostředních přechodů ze stavu, v němž se přechodový systém nachází. Díky této interpretaci lze hovořit také o (přípustných) posloupnostech podmínek.

Důvodem, proč podmínky $\Lambda$ nejsou postačující, je obecnost přechodové relace $R$. Tato relace totiž nemusí být funkcí. Mohou tedy existovat například přechody $(A, \alpha, B)$ a $(A, \alpha, C)$, kde splnění podmínky $\alpha$ (samozřejmě také výskyt ve stavu $A$) nepostačuje k určení, který z obou přechodů nastane. Avšak v rámci přechodového systému nemáme nic dalšího, co by k určení pomohlo. Nastávající přechod (a potažmo nový stav) proto musí být dourčen něčím mimo přechodový systém, což je zároveň zdrojem nedeterminismu přechodového systému.

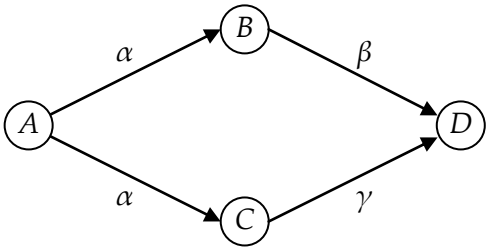
**Charakteristické vlastnosti**

Ačkoliv zevrubným studiem právě definovaného přechodového systému se budeme zabývat později (viz oddíly 2.3 a 2.4), tak už zde si povšimněme jeho podstatných rysů.

Již ze samotné definice vyplynulo, že přechodový systém je obecně nedeterministický. Avšak jak se nedeterminismus projevuje? Jak ovlivňuje vlastnosti přechodového systému? Dobrou představu nám poskytne následující příklad. Mějme tedy přechodový systém $(S, \Lambda, R)$ daný množinami

$$S = \{ A, B, C, D \} \qquad \Lambda = \{ \alpha, \beta, \gamma \} \qquad R = \{ (A, \alpha, B), (A, \alpha, C), (B, \beta, D), (C, \gamma, D) \}$$

a pro větší přehlednost uveďme i jeho přechodový graf




Nyní předpokládejme, že daný přechodový systém je ve stavu $A$. Potom je $\alpha$ jedinou podmínkou, která smí být splněna. A jen a právě jejím splněním dojde k přechodu, resp. dojde buď k přechodu $(A, \alpha, B)$, anebo k přechodu $(A, \alpha, C)$. Projevem nedeterminismu je tedy nemožnost předpovědět (na základě znalosti o splnění podmínky) nový stav a potažmo ani nově splnitelné podmínky. Avšak to neznamená jakoukoliv neurčitost ani stavu, ani splnitelných (přípustných) podmínek, obojí je určeno vždy.

Splněním podmínky $\alpha$ se daný přechodový systém ocitne buď právě ve stavu $B$, anebo právě ve stavu $C$ (jen nelze předpovědět, který z nich to bude). Řekněme tedy, že je ve stavu $B$. Potom smí být splněna (pouze) podmínka $\beta$. Posloupnost podmínek $\alpha\beta$ je tedy přípustná, ale ne vždy. Může se totiž stát, že splněním podmínky $\alpha$ se daný přechodový systém dostane do stavu $C$ a splnění podmínky $\beta$ již nepřipustí, zato by připustil splnění podmínky $\gamma$.

Podstatnou vlastností přechodového systému tedy je, že mohou existovat takové posloupnosti podmínek, o jejichž přípustnosti nelze jednoznačně rozhodnout.

## 2.2 Automat

Pro účely porovnání ovšem musíme zvolit vhodný typ automatu. V rámci teorie automatů je přechodovému systému zřejmě nejbližší nedeterministický konečný poloautomat a proto se s ním stručně seznámíme (obdobně jako s přechodovým systémem v oddílu 2.1).

**Definice nedeterministického konečného poloautomatu**

Přestože nedeterministický konečný poloautomat (dále jen automat) je také abstraktním výpočetním modelem, tak je již dostatečně konkrétní a není nutné ho pro naše potřeby jakkoliv upřesňovat či dodatečně interpretovat. Automat je tedy trojice

$$(Q, \Sigma, \delta)$$

kde $Q$ je konečná množina stavů, $\Sigma$ je konečná množina vstupních znaků a $\delta$ je přechodová funkce. Přesněji řečeno, $\delta : Q \times \Sigma \to 2^Q$, takže můžeme psát

$$\delta(A, \alpha) = U \qquad A \in Q \quad \alpha \in \Sigma \quad U \subseteq Q$$

přičemž platí

$$U \neq \emptyset$$

právě když existuje přechod, resp. potenciální přechody ze stavu $A$ do stavů $U$ podmíněné vstupním znakem $\alpha$. Vzhledem k tomu, že jsou přechody definovány uvedenou funkcí, tak stav automatu nemusí být jednoznačně určen (protože každý přechod vyústí potenciálně do všech stavů z příslušné množiny $U$).

Přesněji řečeno, automat se vždy nachází potenciálně ve všech stavech vymezených nějakou neprázdnou množinou $P \subseteq Q$ (takže ve smyslu množiny $Q$ je stav automatu obecně nedeterministický). S každým převzetím znaku $\alpha \in \Sigma$ ze vstupu pak můžeme s určitostí stanovit pouze novou množinu

$$P' = \bigcup_{X \in P} \delta(X, \alpha) \qquad (1)$$

přičemž z požadavku na neprázdnost množiny $P$ vyplývá, že převzetí znaku $\alpha$ ze vstupu je přípustné právě když $P' \neq \emptyset$. Navíc je zřejmé, že pro každou množinu $P$ (tzn. souhrnně pro kterékoliv stavy, v nichž se automat potenciálně nachází) je možné ze vstupu převzít nejvýše jeden znak. Přirozeně tedy lze hovořit o (přípustných) posloupnostech vstupních znaků.

**Charakteristické vlastnosti**

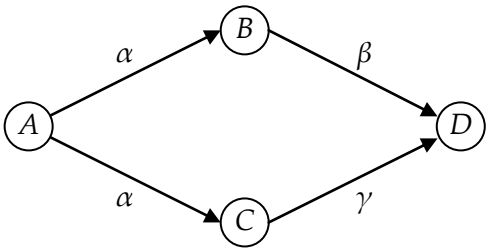
Obdobně jako v případě přechodového systému, také zde si ihned povšimněme podstatných rysů právě definovaného automatu.

Přestože je tento automat označován jako nedeterministický, tak jeho vnější projev (tj. přebírání znaků ze vstupu) je vždy deterministický. Podle vztahu (1) totiž k libovolné množině $P$ a každému vstupnímu znaku $\alpha$ existuje právě jedna množina $P'$. Jinými slovy, ke každému automatu bychom mohli najít deterministický ekvivalent (tzn. deterministický konečný po-

loautomat), který bude mít totožný vnější projev. Tato skutečnost je v teorii automatů dobře známa; jen je dobré si uvědomit, že přechodová funkce takového ekvivalentu by obecně byla funkcí parciální (tzn. nedefinovanou v bodech, kde $P' = \emptyset$). Souhrnně tedy můžeme říci, že nedeterminismus automatu je čistě vnitřní záležitostí. Pro ilustraci uveďme jednoduchý příklad automatu $(Q, \Sigma, \delta)$, kde

$$Q = \{ A, B, C, D \} \qquad \Sigma = \{ \alpha, \beta, \gamma \} \qquad \delta(A, \alpha) = \{ B, C \} \qquad \delta(B, \beta) = \delta(C, \gamma) = \{ D \}$$

a pro větší přehlednost uveďme i jeho přechodový graf




Nyní předpokládejme, že daný automat je ve stavu $A$. Pak je $\alpha$ jediným znakem, který smí být převzat ze vstupu. A jen a právě jeho převzetím dojde ke dvěma potenciálním přechodům. Nedeterminismus automatu spočívá v možnosti ponechat jeho stav neurčený (automat nejen nepotřebuje, ale ani neumožňuje, aby ho cokoliv mimo něj dourčovalo). Proto se daný automat potenciálně ocitne jak ve stavu $B$, tak ve stavu $C$. Potom smí být ze vstupu převzat buď znak $\beta$, anebo znak $\gamma$. Posloupnost vstupních znaků $\alpha\beta$ je tedy přípustná, a to vždy.

Podstatnou vlastností automatu tedy je, že o přípustnosti libovolné posloupnosti vstupních znaků lze jednoznačně rozhodnout (např. na základě existence alespoň jednoho odpovídajícího sledu potenciálních přechodů v přechodovém grafu).

## 2.3 Vnější pohled a ekvivalence

Pro účely porovnání nyní ztotožníme podmínky přechodového systému se vstupními znaky automatu. Přesněji řečeno stanovíme, že podmínka $\alpha \in \Lambda$ přechodového systému je splněna, právě když je stejnojmenný znak $\alpha \in \Sigma$ přechodovým systémem převzat ze vstupu. Analogicky bude platit, že převzetí vstupního znaku je přípustné, právě když je přípustné splnění stejnojmenné podmínky. V důsledku lze tedy přechodový systém (studovaný v této kapitole) vyjádřit trojicí $(S, \Sigma, R)$, což nám umožní přehlednější porovnání s automatem $(Q, \Sigma, \delta)$.

Protože přechodový systém budeme s automatem porovnávat z vnějšího i vnitřního pohledu, je třeba si vymezit rozhraní, přes které by na sebe oba pohledy navazovaly a které by bylo shodné pro přechodový systém i automat. Prozatím rozhraní naznačíme těmito intuitivními pojmy: vstupní znak, přípustnost převzetí vstupního znaku, přípustná posloupnost vstupních znaků, převzetí vstupního znaku a převzatá posloupnost vstupních znaků. Později (viz oddíl 2.4) již budeme moci rozhraní upřesnit.

**Vnější pohled**

Z vnějšího pohledu má přechodový systém s automatem mnoho společného. V obou případech, po ztotožnění podmínek a (vstupních) znaků, existuje konečná množina znaků

$$\Sigma$$

které mohou být (vždy jen postupně) přebírány přes zmíněné rozhraní. Nejpozději na rozhraní se tak utváří nějaká posloupnost znaků. V obou případech také platí, že jakýkoliv znak nemusí být převzat kdykoliv. Přesněji řečeno, vždy existuje množina znaků

$$\Phi \subseteq \Sigma$$

pouze jejichž bezprostřední převzetí je přípustné. Přitom množina $\Phi$ se s každým převzetím znaku může změnit. Existují tedy přípustné posloupnosti znaků, tj. množina posloupností

$$\Pi \subseteq \Sigma^{\infty} \qquad\qquad \Sigma^{\infty} = \Sigma^{*} \cup \Sigma^{\omega}$$

ve kterých mohou znaky projít přes rozhraní. A konečně, v obou případech může nastat, že množina $\Phi$ je prázdná. Tehdy, ale i nadále, nebude převzat žádný znak (resp. množina $\Phi$ je prázdná $\Rightarrow$ nebude převzat žádný znak $\Rightarrow$ množina $\Phi$ se nezmění, tj. zůstane prázdná). Na druhou stranu, množina $\Phi$ také může být vždy neprázdná a pak neexistuje žádná libovolně dlouhá posloupnost znaků, po níž by nemohl být převzat další znak. Přípustné posloupnosti znaků tedy mohou být jak konečné ($\Sigma^{*}$), tak nekonečné ($\Sigma^{\omega}$).

V čem se (z vnějšího pohledu) přechodový systém od automatu liší? Rozdíl spočívá pouze ve změnách množiny $\Phi$, ke kterým dochází právě s převzetím znaku. Řekněme, že

$$\varphi \in \Sigma^{*}$$

je posloupnost všech již převzatých znaků (jedná se tedy o libovolně dlouhý prefix některé z přípustných posloupností) a předpokládejme, že můžeme experimentovat (tj. pokusit se o převzetí posloupnosti $\varphi$ opakovaně se stejnými počátečními podmínkami).

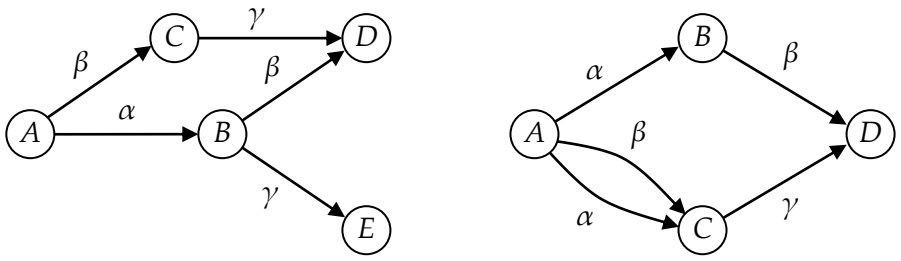
V případě automatu vždy existuje závislost $\Phi(\varphi)$, tj. existuje nějaká funkce uvádějící množinu $\Phi$ do závislosti na posloupnosti $\varphi$. Po opakovaném převzetí posloupnosti $\varphi$ tedy znak $\alpha \in \Phi(\varphi)$ bude převzat vždy a naopak znak $\beta \notin \Phi(\varphi)$ nebude převzat nikdy. Výše jsme mohli konstatovat pouze to, že přípustné mohou být jen některé posloupnosti znaků (zahrnuté do množiny $\Pi$). Nyní však můžeme doplnit, že u automatu je rozdělení posloupností znaků na přípustné a nepřípustné určeno ostře. Jinými slovy, posloupnost znaků je buď vždy přípustná, anebo vždy nepřípustná.

Naproti tomu v případě přechodového systému závislost $\Phi(\varphi)$ obecně neexistuje, resp. množina $\Phi$ může záviset také (či dokonce zcela) na něčem jiném než je posloupnost $\varphi$. Řekněme pro názornost, že množinu $\Phi$ po výchozím převzetí posloupnosti $\varphi$ označíme jako $\Phi'$ a po opakovaném převzetí téže posloupnosti ji označíme jako $\Phi''$. A pokud závislost $\Phi(\varphi)$ neexistuje, tak množiny $\Phi'$ a $\Phi''$ mohou být rozdílné. Po opakovaném převzetí posloupnosti $\varphi$ tedy znak $\alpha \in \Phi'$ bude převzat tehdy a jen tehdy, když $\alpha \in \Phi''$ (analogicky pro znak $\beta \notin \Phi'$ a jeho nepřevzetí). V důsledku se tedy opakované převzetí (celé) posloupnosti $\varphi$ ani nemusí podařit. Z uvedeného je zřejmé, že rozdělení posloupností znaků na přípustné a nepřípustné není

u přechodového systému určeno ostře. Přesněji řečeno, posloupnost znaků je buď vždy přípustná, anebo potenciálně přípustná, anebo vždy nepřípustná.

Nakonec se ještě vrátíme k jednomu výše uvedenému tvrzení. Řekli jsme, že u přechodového systému může množina $\Phi$ záviset také (či dokonce zcela) na něčem jiném než je posloupnost $\varphi$. Ale na čem jiném? Vždyť z vnějšího pohledu působí na přechodový systém pouze vstupní znaky a proto by posloupnost $\varphi$ měla reprezentovat vše co se s přechodovým systémem událo. Avšak nereprezentuje. Množina $\Phi$ tak minimálně z vnějšího pohledu může záviset i na něčem neurčitém, resp. její závislost na posloupnosti $\varphi$ může být nedeterministická. Takovou závislost označme jako $\Phi[\varphi]$ pro snazší odlišení od deterministické $\Phi(\varphi)$. Všimněme si, že zde vypozorovaný nedeterminismus je vlastně již dříve zmiňovaným nedeterminismem (viz oddíl 2.1), který se jen a právě prostřednictvím závislosti $\Phi[\varphi]$ promítá do vnějšího projevu přechodového systému.

Nyní stručně shrneme ty nejdůležitější vlastnosti, ke kterým jsme zatím dospěli. V obou případech (tj. jak u přechodového systému, tak u automatu) existuje množina $\Pi$ právě takových konečných i nekonečných posloupností, v nichž je převzetí vstupních znaků vůbec přípustné. Dále, v obou případech lze zaznamenat posloupnost $\varphi$ těch vstupních znaků, které byly skutečně převzaty. A konečně, v obou případech také existuje množina $\Phi$ takových vstupních znaků, které jsou přípustné bezprostředně (tzn. jako další znak posloupnosti $\varphi$). Přitom množina $\Phi$ se může změnit po každém převzetí vstupního znaku (tzn. v nějaké závislosti na posloupnosti $\varphi$). Avšak závislost množiny $\Phi$ na posloupnosti $\varphi$ je v případě automatu (zcela) deterministická, naproti tomu v případě přechodového systému může tato závislost obsahovat nedeterminismus. Proto posloupnosti množiny $\Pi$ jsou u automatu přípustné vždy, avšak u přechodového systému mohou být některé (nebo všechny) přípustné jen potenciálně. Navíc, ani při znalosti množiny $\Pi$ (resp. bez jakékoliv znalosti o vnitřní struktuře přechodového systému) nelze rozlišit posloupnosti vždy přípustné od potenciálně přípustných. Snadno si to můžeme ukázat na příkladu dvou přechodových systémů, jejichž vnitřní strukturu vystihují následující přechodové grafy




$$\Pi = \{ \beta\gamma, \alpha\beta, \alpha\gamma \} \qquad (2)$$

a jejichž množina přípustných posloupností $\Pi$ je totožná. Přitom posloupnost $\beta\gamma$ je v obou případech přípustná vždy, avšak posloupnosti $\alpha\beta$ a $\alpha\gamma$ jsou v druhém případě přípustné jen potenciálně. Tento příklad nás nutně přivádí k otázce, jak je to s ekvivalencí.

**Ekvivalence**

Ekvivalencí se budeme zabývat ve smyslu jednak vztahu dvou přechodových systémů nebo automatů a jednak vztahu přechodového systému a automatu.

V případě dvou automatů je záležitost vcelku jednoduchá. Automaty jsou ekvivalentní, právě když jsou jejich množiny $\Pi$ shodné. Stručně si to ukážeme následující úvahou. Mějme dva automaty, jejich shodnou množinu $\Pi$ a také dvě libovolné posloupnosti $\xi \notin \Pi$, $\zeta \in \Pi$. Potom převzetí posloupnosti $\xi$ žádný z obou automatů nikdy nepřipustí, což je jasné. Nezanedbatelné však je, že oba automaty odmítnou posloupnost $\xi$ převzít vždy na stejném místě (tj. po stejném počtu již převzatých znaků), a to právě díky determinismu závislosti $\Phi(\varphi)$. Naopak převzetí posloupnosti $\zeta$ oba automaty připustí. Přičemž zde je důležité, že jej připustí vždy, a to opět díky determinismu závislosti $\Phi(\varphi)$. Z uvedeného je pak zřejmé, že žádnou posloupností vstupních znaků nelze oba automaty rozlišit. A proto ze shodnosti množin $\Pi$ vyplývá ekvivalence automatů. Platnost implikace v opačném směru (tj., že shodnost množin $\Pi$ vyplývá z ekvivalence, resp. nerozlišitelnosti automatů) již lze snadno ověřit sporem.

V případě dvou přechodových systémů je ekvivalence již složitější záležitost. Z předchozího příkladu (2) je patrné, že shodnost množin $\Pi$ není pro ekvivalenci přechodových systémů postačující podmínkou. Důvodem je, že pokud daným přechodovým systémům předložíme například posloupnost $\alpha\beta \in \Pi$, tak tím levým bude převzata vždy, avšak tím pravým může být převzetí odmítnuto na znaku $\beta$, a to právě kvůli nedeterminismu závislosti $\Phi[\varphi]$. Existuje tedy posloupnost vstupních znaků, kterou lze dané přechodové systémy rozlišit. A proto ze shodnosti množin $\Pi$ ekvivalence přechodových systémů (obecně) nevyplývá.

Je tedy třeba najít výstižnější formální popis vnějšího projevu, vhodný pro posouzení ekvivalence přechodových systémů. Ihned se nabízí nedeterministická závislost $\Phi[\varphi]$, bude ji však možné vhodným způsobem formálně vyjádřit? Nebo bude stačit již samotná vnitřní struktura přechodových systémů? Anebo bude vhodnější najít jiný formální popis? Problematika se však stává ještě složitější. Je třeba navíc vyjasnit, co ekvivalencí vlastně míníme, neboť nedeterminismus přechodových systémů s sebou přináší i jemnější odlišnosti ve vnějším projevu, než je samotná potenciální přípustnost posloupností vstupních znaků. Jsou třeba ekvivalentní dva přechodové systémy, které nějakou nepřípustnou posloupnost odmítnou převzít sice vždy, ale nikoliv na shodném místě (tj. po shodném počtu již převzatých znaků)? Zde naznačenou problematikou se budeme podrobněji zabývat dále (zejména pak v kapitole 5).

Nyní se krátce podíváme na ekvivalenci přechodového systému a automatu, resp. na vztah mezi těmito dvěma abstraktními výpočetními modely. Z dosud uvedeného je zcela zjevné, že přechodové systémy a automaty obecně nejsou z vnějšího pohledu ekvivalentní. Avšak také jsme dospěli ke zjištění, že rozdíl spočívá pouze v nedeterminismu závislosti množiny $\Phi$ na posloupnosti $\varphi$. Znamená to tedy, že deterministické přechodové systémy (tzn. takové, kde zmíněná závislost je deterministická) již s automaty ekvivalentní jsou. Jinými slovy, obecně jsou automaty z vnějšího pohledu podmnožinou přechodových systémů.

Nakonec si všimneme podobnosti množiny $\Pi$ s množinou *traces*. Podle klasické definice, se kterou pracuje třeba Hoare [4], je *traces* množina všech možných (resp. zaznamenatelných) posloupností, ve kterých by přechodový systém či automat převzal vstupní znaky. Přesněji řečeno, $traces \subseteq \Sigma^*$ je množina všech možných posloupností $\varphi$, a to včetně prázdné posloupnosti $\varepsilon$. Stejně jako množina $\Pi$, ani *traces* nerozlišuje, ve kterých posloupnostech přechodový systém vstupní znaky převezme vždy a ve kterých pouze potenciálně. Avšak na rozdíl od $\Pi$, množina *traces* nedokáže u některých posloupností rozlišit, zda přechodový systém po konečném (byť libovolně vysokém) počtu převzatých vstupních znaků může převzetí dalšího odmítnout, anebo bude schopen vstupní znaky přebírat neustále. Podle jiné definice, s níž

pracuje například Broy [3], je množina *traces* (obvykle označovaná jako *completed traces*) definována téměř shodně s naší množinou $\Pi$. Jediný přetrvávající rozdíl potom spočívá v tom, že množina $\Pi$ nikdy neobsahuje prázdnou posloupnost $\varepsilon$. Například pro přechodový systém nepřipouštějící převzetí žádného vstupního znaku tak dostaneme $\Pi = \emptyset$, ale *traces* = { $\varepsilon$ }.

Ačkoliv absence prázdné posloupnosti v množině $\Pi$ je z hlediska samotné ekvivalence zdánlivě nepodstatný rozdíl, tak později (viz kapitola 4) by nám existence jakýchkoliv prázdných posloupností naopak spolehlivě zabránila vybudovat algebraickou strukturu.

## 2.4 Vnitřní pohled

Nejdříve si připomeňme, že na začátku předchozího oddílu jsme pro účely porovnání ztotožnili podmínky přechodového systému se vstupními znaky automatu. Dále jsme vymezili rozhraní mezi vnějším a vnitřním pohledem, byť jen intuitivními pojmy. Nyní, na základě poznatků z porovnání přechodového systému a automatu z vnějšího pohledu, však již můžeme rozhraní upřesnit, a to ve formě čtveřice následujících symbolů:

- $\Sigma$ – konstantní, konečná množina všech vstupních znaků (tj. veškerých znaků, které kdy mohou být převzaty přes toto rozhraní). Nutno podotknout, že množina $\Sigma$ neobsahuje nadbytečné znaky (resp. množiny, kterých $\Phi$ nabývá, musí tvořit pokrytí množiny $\Sigma$).

- $\Phi$ – proměnná, konečná množina takových vstupních znaků, pouze jejichž bezprostřední převzetí je přípustné (přičemž $\Phi \subseteq \Sigma$). Ovšem k významu množiny $\Phi$ je třeba dodat jednak, že vstupní znaky je možné přebírat vždy jen postupně. Dále, že množina $\Phi$ se může změnit jen a právě s každým převzetím znaku. A konečně, že množina $\Phi$ může být prázdná. Vnitřní struktura musí být vždy schopna určit množinu $\Phi$, ale jak ji má určit už rozhraní nepředepisuje. Pouze připouští, že $\Phi$ může záviset také (či zcela) na posloupnosti $\varphi$.

- $\Pi$ – konstantní, obecně nekonečná množina všech (tj. jak potenciálně, tak vždy) přípustných posloupností vstupních znaků (přičemž $\Pi \subseteq \Sigma^\infty$). Existence této množiny omezuje variabilitu množiny $\Phi$, resp. ukládá vnitřní struktuře přinejmenším neměnit $\Phi$ zcela libovolně. Kdybychom vyloučili potenciálně přípustné posloupnosti, pak $\Pi$ dokonce plně předepisuje jak se má $\Phi$ měnit. Naopak, pokud by se $\Phi$ měnila náhodně, tak by byla potenciálně přípustná každá posloupnost množiny $\Sigma^\infty$.

- $\varphi$ – proměnná, konečná posloupnost všech dosud převzatých vstupních znaků (přičemž platí, že $\varphi\zeta \in \Pi$, kde $\varphi \in \Sigma^*$ a $\zeta \in \Sigma^\infty$, tj. že posloupnost $\varphi$ je libovolně dlouhým prefixem některé z posloupností množiny $\Pi$). Existence posloupnosti $\varphi$ znamená, že z vnějšího pohledu je vždy rozlišitelná situace před a po převzetí každého znaku (tzn. můžeme také hovořit o vstupní historii $\varphi$).

Z vnitřního pohledu mají přechodový systém ($S$, $\Sigma$, $R$) a automat ($Q$, $\Sigma$, $\delta$) společné pouze uvedené rozhraní. V tom ostatním (tj. ve vnitřní struktuře dané prostřednictvím $S$, $R$ a $Q$, $\delta$) se liší. Jinými slovy, rozdíly hledejme v definicích stavů a přechodů.

**Stav**

Množiny stavů $S$ a $Q$ se mohou lišit i ve velikosti a přesto může být vnější projev příslušného přechodového systému i automatu ekvivalentní, posuzování samotných množin stavů nám tedy nepomůže. Podstatný je totiž rozdíl ve významu samotného pojmu stav.

V případě automatu lze pojem stav definovat jako důsledek veškeré možné vstupní historie (přitom u deterministického automatu můžeme dokonce stav definovat přímo jako veškerou možnou vstupní historii). Abychom to však mohli ukázat, musíme si napřed definovat zobecněnou přechodovou funkci

$$\delta^*(P,\xi) \;=\; \begin{cases} P & \ldots\text{pokud } \xi \text{ je prázdná posloupnost} \\ \delta^*\left(\bigcup\limits_{X\in P}\delta(X,\alpha)\,,\quad \zeta\right) & \ldots\text{pokud } \xi=\alpha\zeta,\ \alpha\in\Sigma,\ \zeta\in\Sigma^* \end{cases}$$

pomocí které zavedeme relaci ekvivalence ~ nad množinou $\Sigma^*$ (tj. nad množinou všech konečných posloupností vstupních znaků)

$$\xi \sim \zeta \quad\Leftrightarrow\quad \delta^*(P_1,\xi) \;=\; \delta^*(P_1,\zeta)$$

přičemž $P_1 \subseteq Q$ je množinou nějakých počátečních stavů. Relace ekvivalence ~ indukuje rozklad $\Sigma^*/_\sim = \{\ T(\xi) : \xi \in \Sigma^*\ \}$ na třídy $T(\xi) = \{\ \zeta : \zeta \sim \xi\ \}$ a pokud pro každou třídu $T(\xi_i)$, resp. pro jejího reprezentanta $\xi_i$ označíme $P_i = \delta^*(P_1, \xi_i)$, potom zřejmě existuje bijektivní zobrazení mezi rozkladem $\Sigma^*/_\sim$ a pokrytím $\{\ P_i : P_i \subseteq Q\ \}$. Význam tohoto zobrazení můžeme vyjádřit (zjednodušeně zapsaným) tvrzením

$$nastala(\ T(\xi_i)\ ) \;\Leftrightarrow\; nastaly(\ P_i\ )$$

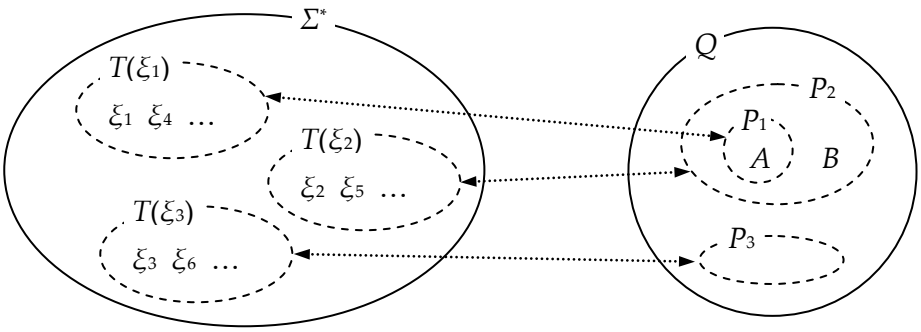
kde $T(\xi_i)$ představuje veškerou možnou vstupní historii, která podle zmiňovaného zobrazení odpovídá množině potenciálních stavů $P_i$, v nichž by se daný automat právě s takovou možnou historií nacházel. Pro jednotlivé potenciální stavy pak zřejmě platí slabší tvrzení

$$nastala(\ T(\xi_i)\ ) \;\Rightarrow\; nastal(\ X\ ) \qquad X \in P_i \tag{3}$$

a tím je také ukázána správnost výše předeslané definice pojmu stav, která chápe stav automatu jako důsledek jeho veškeré možné vstupní historie. Tvrzení (3) pro úplnost rozepíšeme podrobněji na

$$\forall\ \xi_i \in \Sigma^*,\ X \in \delta^*(P_1,\xi_i) \quad :\quad \varphi \in T(\xi_i) \;\Rightarrow\; X \in \delta^*(P_1,\varphi)$$

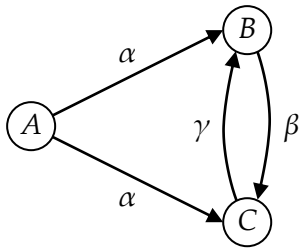
čímž jsme zároveň odkryli návaznost tvrzení (3) a potažmo definice stavu na dříve zvolené rozhraní mezi vnějším a vnitřním pohledem, konkrétně na proměnnou $\varphi$ představující (skutečně nastalou) vstupní historii. Pro větší názornost ještě uvedeme obrázek, který (pro nějaký konkrétní automat) ilustruje popisované množiny a vztahy mezi nimi, včetně zmiňovaného zobrazení vyznačeného šipkami



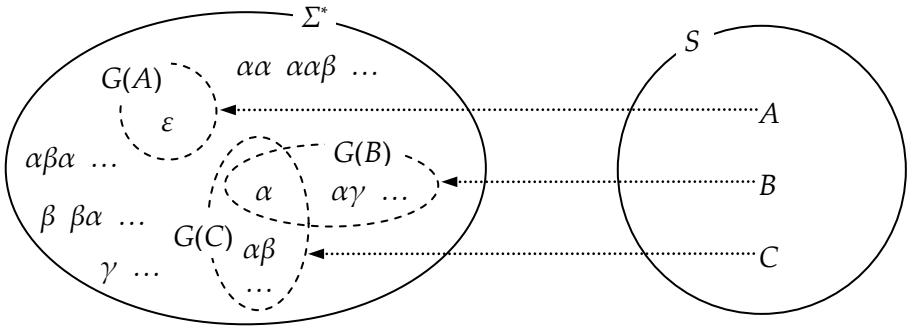

V případě přechodového systému však pojem stav definovat jako důsledek veškeré možné vstupní historie nelze. A protože tvrdíme, že to obecně nelze, tak stačí když na jednom konkrétním příkladu ukážeme spor. Mějme tedy přechodový systém $(S, \Sigma, R)$, kde

$$S = \{ A, B, C \} \qquad \Sigma = \{ \alpha, \beta, \gamma \} \qquad R = \{ (A, \alpha, B), (A, \alpha, C), (B, \beta, C), (C, \gamma, B) \}$$

a uveďme i jeho přechodový graf




ke kterému zřejmě můžeme nalézt relaci dosažitelnosti $\xi \to X$ vyjadřující, že existuje sled přechodů označených vstupními znaky (v pořadí) posloupnosti $\xi \in \Sigma^*$, který vede z počátečního stavu $A$ do stavu $X \in S$. Potom zřejmě také můžeme zavést množiny $G(X) \subseteq \Sigma^*$, přesněji $G(X) = \{ \xi : \xi \to X \}$. Konkrétně lze celou situaci znázornit obrázkem




přičemž veškerá možná vstupní historie přechodového systému nacházejícího se ve stavu $X$ je představována právě množinou $G(X)$. Předpokládejme tedy, že platí tvrzení

$$nastala(\, G(X)\,) \;\Rightarrow\; nastal(\, X\,)$$

které můžeme rozepsat podrobněji na

$$\forall\; X \in S \quad : \quad \varphi \in G(X) \;\Rightarrow\; nastal(\, X\,) \tag{4}$$

kde stojí za povšimnutí ponechaný predikát *nastal*. Tento predikát totiž podrobněji rozepsat neumíme, což koresponduje s dřívějším zjištěním (viz oddíl 2.1), že nastávající stav může být dourčován něčím mimo přechodový systém. Jinými slovy, ani znalost vnitřní struktury ($S$, $\Sigma$, $R$) ani znalost rozhraní ($\Sigma$, $\Phi$, $\Pi$, $\varphi$) přechodového systému nedostačuje k vyjádření tohoto predikátu. Pro ukázání slibovaného sporu to však překážkou nebude. V právě zkoumaném konkrétním příkladu totiž podle tvrzení (4) musí být splněny i následující dva výroky

$$\alpha \in G(B) \;\Rightarrow\; nastal(\, B \,)$$

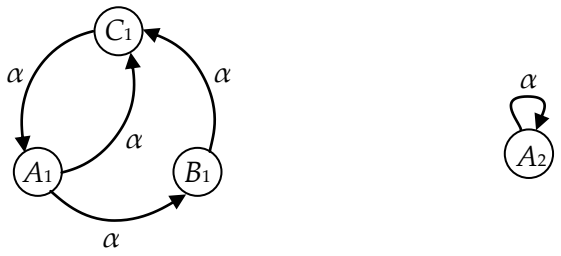
$$\alpha \in G(C) \;\Rightarrow\; nastal(\, C \,)$$

takže musí být splněn jak výrok *nastal*( $B$ ), tak výrok *nastal*( $C$ ). Něco takového je však ve sporu s definicí přechodového systému (viz oddíl 2.1), podle které je stav vždy jednoznačně určen. Proto tvrzení (4) platné být nemůže a stav přechodového systému opravdu nelze definovat jako důsledek veškeré možné vstupní historie. Vyvstává tedy otázka, co znamená pojem stav u přechodových systémů a jak bychom ho mohli definovat? Dva možné způsoby si ukážeme později (viz kapitoly 3 a 4).

**Přechody**

Vzhledem k tomu, že veškeré přechody jsou jasně definovány na jedné straně relací $R$ a na druhé funkcí $\delta$, tak z formálního hlediska je rozdíl mezi přechodovým systémem a automatem zřejmý. Přímým důsledkem je rozdílný charakter jejich nedeterminismu (podstatných rysů jsme si všimli již v oddílech 2.1 a 2.2, vliv na vnější projev a ekvivalenci byl pak diskutován v oddílu 2.3).

Obraťme proto pozornost opět k významu a na následujícím příkladu prozkoumejme vyjadřovací schopnost nikoliv jednotlivých přechodů, ale celé struktury přechodů. Mějme dva přechodové grafy




(5)

přičemž každý z nich může být interpretován jako přechodový systém nebo automat (dostaneme tedy čtyři různé interpretace). Intuitivně lze ihned odhadnout, že všechny interpretace budou vzájemně ekvivalentní. Přesto si tuto domněnku formálně ověříme.

S využitím výše zavedených množin $T$ a $G$ lze pro všechny čtyři interpretace zjistit veškeré možné vstupní historie. A protože nám zde nezáleží na výsledném stavu, v němž se přechodový systém nebo automat s určitou vstupní historií nachází, tak dostaneme

$$T_1(\varepsilon) \;\cup\; T_1(\alpha) \;\cup\; T_1(\alpha\alpha) \;\cup\; T_1(\alpha\alpha\alpha) \;=\; \Sigma^*$$

$$G_1(A_1) \;\cup\; G_1(B_1) \;\cup\; G_1(C_1) \;=\; \Sigma^*$$

$$T_2(\varepsilon) = \Sigma^*$$

$$G_2(A_2) = \Sigma^*$$

kde $\Sigma^* = \{ \varepsilon, \alpha, \alpha\alpha, \alpha\alpha\alpha, \alpha\alpha\alpha\alpha, \dots \}$. Veškerá možná vstupní historie (bez ohledu na výsledný stav) je tedy u všech čtyř interpretací totožná, tzn. u všech je možné z vnějšího pohledu zpozorovat tutéž vstupní historii $\varphi \in \Sigma^*$. Pokud bychom uvažovali jen automaty, tak by již toto zjištění postačovalo k vynesení soudu o ekvivalenci. Avšak uvažujeme i přechodové systémy a díky poznatkům získaným z vnějšího pohledu (viz oddíl 2.3) víme, že budoucí vnější projev nezávisí pouze na pozorovatelné historii. Přesněji víme, že musíme ověřit také shodnost závislostí množiny $\Phi$ (tj. množiny bezprostředně přípustných vstupních znaků) na vstupní historii $\varphi$. V tomto případě je to triviální, jako (deterministické i nedeterministické) závislosti pro všechny čtyři interpretace dostaneme

$$\Phi_1(\varphi) = \Phi_1[\varphi] = \Phi_2(\varphi) = \Phi_2[\varphi] = \{ \alpha \} \qquad \varphi \in \Sigma^*$$

což znamená, že po libovolné vstupní historii je u všech čtyř interpretací vždy přípustné bezprostřední převzetí právě a jen vstupního znaku $\alpha$. Výsledky právě stručně popsaného formálního ověření samozřejmě nejsou překvapující, avšak přísně vzato, teprve nyní můžeme s jistotou považovat všechny čtyři interpretace za ekvivalentní, resp. z vnějšího pohledu nerozlišitelné (neboť ekvivalence může být chápána i jemněji, viz oddíl 5.3).

Přestože jsou oba přechodové grafy (5) ve všech popsaných interpretacích ekvivalentní, tak grafy samotné (potažmo i odpovídající přechodové relace $R_1$, $R_2$ či funkce $\delta_1$, $\delta_2$) jsou podstatně odlišné. Jak přechodový graf, tak vnitřní struktura $(S, \Sigma, R)$ přechodového systému nebo $(Q, \Sigma, \delta)$ automatu tedy vyjadřují mnohem více než jen příslušný vnější projev. V rámci teorie automatů je tato skutečnost známa, ostatně stačí připomenout pojmy jako minimální automat, ekvivalentní stavy nebo isomorfismus automatů. Avšak i vnitřní struktura minimálního automatu stále ještě nese více informace, než je nezbytné k vyjádření jeho vnějšího projevu (ostatně pokud by tomu tak nebylo, nepotřebovali bychom zavádět zmíněný isomorfismus automatů). Obdobné je to také u přechodových systémů, kde situaci navíc komplikuje jejich nedeterminismus (jenž se, na rozdíl od automatů, promítá i do vnějšího projevu). Jinak řečeno, pro formální popis (pouze a právě) vnějšího projevu přechodových systémů bychom tedy potřebovali něco sice výstižnějšího než jejich množinu $\Pi$, avšak přitom méně obsažného než jejich vnitřní strukturu. Dva možné kandidáty na takovýto popis si ukážeme později (viz kapitoly 3 a 4).

Na vyvstávající otázku co dalšího (kromě vnějšího projevu) přechody tedy vyjadřují asi nelze odpovědět zcela obecně. Alespoň pro představu se podívejme na všeobecně známou aplikaci abstraktní algebry na automaty, v níž přechody vyjadřují transformace (viz třeba Holcombe [6] či Thomas [5]). Je totiž možné ukázat, že každému vstupnímu znaku $\alpha \in \Sigma$ odpovídá právě jedna transformace množiny tříd $\Sigma^*/_\sim$, která je definována právě přechody označenými daným vstupním znakem $\alpha$. Dodejme, že transformací rozumíme (obecně parciální) funkci $f : \Sigma^*/_\sim \to \Sigma^*/_\sim$. Každé posloupnosti vstupních znaků potom v důsledku odpovídá právě jedna složená transformace atd. Jednotlivým třídám můžeme přiřadit nějaké pojmy a posloupnostmi vstupních znaků pak můžeme, prostřednictvím vhodně zvoleného automatu, provádět na těchto pojmech libovolné transformace.

Asi tím nejjednodušším příkladem je klasifikace posloupností vstupních znaků. Mějme automat v počátečním stavu a posloupnost vstupních znaků $\xi \in \Sigma^*$. Známe tedy množinu tříd $\Sigma^*/_{\sim}$ a složenou transformaci $f_\xi$. Automat pak převzetím posloupnosti $\xi$ prakticky provede

$$f_\xi(\ T(\varepsilon)\ ) = T(\xi) \qquad (6)$$

což znamená, že automat pro posloupnost $\xi$ prakticky vypočte třídu $T(\xi)$, do které tato posloupnost patří. Jinými slovy, dojde ke klasifikaci této posloupnosti.

Zde je dobré zdůraznit, že i když jsou automaty z vnějšího pohledu nerozlišitelné, tak mohou provádět zcela rozdílné transformace. Pro ilustraci se vraťme ke dvěma přechodovým grafům (5), u kterých jsme potvrdili, že interpretovány (nejen) jako automaty jsou z vnějšího pohledu nerozlišitelné. Přitom automat podle levého přechodového grafu umožňuje rozlišovat (klasifikovat) délky posloupností vstupních znaků (resp. pouze délky 0, 1, 2 a ostatní), avšak automat podle pravého přechodového grafu to nedokáže (jím vyjadřovaná transformace je totiž identita).

Zjistili jsme, že přechody (a tedy i vnitřní struktura jako taková) obecně nesou dodatečnou informaci, na které vnější projev nezávisí (a pro jehož vyjádření je tak nadbytečná). Ovšem také jsme si ukázali, že tato dodatečná informace může vyjadřovat i smysluplné vlastnosti systému (např. zmíněné transformace). Avšak tyto vlastnosti nejsou z vnějšího pohledu pozorovatelné, resp. nemají žádný dopad na okolí, s nímž systém interaguje prostřednictvím výše definovaného rozhraní ($\Sigma$, $\Phi$, $\Pi$, $\varphi$). Tím se opět dostáváme k již výše řečenému, že pro formální popis vnějšího projevu bychom potřebovali něco méně obsažného než vnitřní strukturu (tzn. něco bez zmiňované dodatečné informace).

Navíc je užitečné si všimnout, že v zápisu transformace (6) ve výše uvedeném příkladu klasifikace posloupností jsme se obešli bez stavů, resp. nepotřebovali jsme symboly prvků nějaké množiny $Q$. Tato skutečnost nám už signalizuje, že ona dodatečná informace nespočívá jen v pojmenování stavů, ale především v samotné podstatě struktury přechodů, tzn. v topologii přechodového grafu. Takže formální popis (jen a právě) vnějšího projevu sice musí do nezbytné míry odrážet strukturu přechodů, ale zřejmě by na ní neměl být jakkoliv založen.

# 3 Milnerův základní kalkulus

## 3.1 Souvislosti

Milner [1] pohlíží na dynamický systém jako na kompozici vzájemně na sebe působících částí, které nazývá agenty. Navíc připouští hierarchickou strukturu takového systému, agent tak sám může být systémem složeným z dalších agentů. Pojem agent je tedy široký a „znamená jakýkoliv systém jehož chování spočívá v diskrétních akcích“ [1]. Z uvedeného tedy vyplývá, že agent a systém jsou ekvivalentní pojmy.

Pojmem akce (ve výše uvedené citaci) je míněno právě ono jednotlivé a atomické působení dvou agentů na sebe navzájem. Přitom v rámci systému může nezávisle na sobě (tedy souběžně) proběhnout více akcí, pokud každá z těchto akcí proběhne buď mezi jinou dvojicí agentů, anebo uvnitř jednoho agenta. Milner akce zároveň chápe jako komunikaci (akci uvnitř agenta lze pak chápat jako komunikaci jeho částí) a soudí, že „je tedy rozumné definovat chování systému právě jako veškerou schopnost komunikace; jinými slovy, chování systému je přesně to, co je pozorovatelné a pozorovat systém znamená komunikovat s ním“ [1].

Kompozici agentů pak Milner chápe jako utváření (komunikační) sítě agentů, kde pomyslné spoje mezi agenty jsou tvořeny komplementárními názvy akcí. Přesněji řečeno, pokud dva agenty jsou schopny bezprostředně provést akci, nazvanou u prvního $\alpha$ a u druhého $\overline{\alpha}$, tak mezi nimi implicitně existuje onen pomyslný spoj. Nicméně to ještě neznamená uskutečnění komunikace, resp. provedení akce právě mezi těmito dvěma agenty. Pokud by existoval třetí agent také schopný bezprostředně provést akci nazvanou $\overline{\alpha}$ (analogicky platí i pro $\alpha$), tak by soutěžil s druhým agentem. Pomyslné spoje by existovaly dva, avšak akce by proběhla nedeterministicky buď mezi prvním a druhým, anebo mezi prvním a třetím agentem. Je zřejmé, že na provádění akce se oba agenty účastní vždy synchronně, což Milner z komunikačního hlediska vystihuje tzv. handshake (tedy synchronní) komunikací.

Při budovaní kalkulu nejprve Milner zjednodušuje komunikaci na pouhou synchronizaci, při které nastává jen samotný handshake bez přenosu jakékoliv informace. Tento přístup je pozoruhodný, neboť tak dospěje k dostatečně ucelenému tzv. základnímu kalkulu, který v sobě již zahrnuje vše podstatné (postačuje třeba i pro posuzování ekvivalence). Přitom k rozšíření základního kalkulu o přenos informace pak zcela postačují jeho vlastní výrazové prostředky.

Výrazům (nejprve čistě syntakticky budovaného) základního kalkulu Milner přikládá význam teprve dodatečně prostřednictvím přechodových systémů (jakým způsobem uvidíme později), které přinášejí i pojem stav. Ve snaze zjednodušit terminologii proto nadále ztotožňuje pojmy agent a stav. V rámci kalkulu tedy „pojmy *agent* i *stav* budou vždy chápány jako agent (systém) nacházející se ve *stavu*“ [1]. Nicméně takováto definice je poměrně zavádějící, protože *agent* (*stav*) se nemůže vyvíjet souběžně s jiným *agentem* (*stavem*) v rámci stejného systému, kdežto dva agenty (systémy) samozřejmě mohou. Pojmům *agent* i agent se proto v následujících odstavcích již vyhneme.

Vzhledem k tomu, že spíše než využití kalkulu nás bude zajímat jak je kalkulus vybudován, dále jaké má v důsledku toho vlastnosti a později také co (na modelovaném systému) vlastně modeluje (s čímž třeba úzce souvisí, jaké systémy jsou v rámci kalkulu chápány jako ekvivalentní), tak se zaměříme na samotnou podstatu kalkulu. Stručně řečeno, v rámci základního kalkulu nás bude zajímat modelování právě a jen jednoho systému. Ostatní systémy, které se prostřednictvím akcí synchronizují s tímto systémem, tedy splynou v jeho okolí. Jinými slovy, ze základního kalkulu vynecháme kompozici systémů a vše co s ní souvisí (tedy třeba i výše zmiňované komplementární názvy akcí – zkrátka akce $\bar{\alpha}$ bude chápána již pouze jako nějaká akce různá od akce $\alpha$, můžeme ji pak nazývat třeba $\beta$). Tím dostaneme určitý výřez základního kalkulu, který plně postačuje našemu zmiňovanému zaměření. V dalších odstavcích si stručně popíšeme, resp. zrekonstruujeme Milnerův základní kalkulus v rámci právě zvoleného výřezu.

## 3.2 Jazyk a jeho syntaxe

Naši rekonstrukci základního kalkulu (v rámci zvoleného výřezu) rozdělíme, tak jako Milner [1], do dvou částí. Nejdříve definujeme syntaxi jazyka, v němž jsou zapisovány výrazy kalkulu. A následně dodefinujeme význam slov tohoto jazyka, tj. význam výrazů kalkulu.

Mějme tedy množinu akcí *Act* a množinu konstant *K*, potom množina výrazů $\mathcal{Z}$ je rekurzivně definována pouhými třemi generickými výrazy, resp. předpisem

$$\mathcal{Z} = \{\ A,\ \alpha\cdot(E),\ \textstyle\sum_{i\in I}(E_i)\ :\ A \in K; \alpha \in Act; E, E_i \in \mathcal{Z}; I \subseteq \boldsymbol{N}\ \} \qquad (7)$$

Implicitně tak byly zavedeny operace · a + bez rozlišení jejich priority. Kvůli redukci počtu závorek v zápisech výrazů tedy dodejme, že operace · bude mít vyšší prioritu. O vlastnostech operací (asociativita, komutativita, existence neutrálního prvku apod.) však nyní nelze nic říci.

Všimněme si, že množina indexů *I* v uvedeném předpisu (7) smí být prázdná. Tím byl (opět implicitně) zaveden tzv. prázdný výraz, který budeme zkráceně zapisovat symbolem **0**. Pro ilustraci si ukažme několik zápisů generického výrazu $\sum_{i\in I}(E_i)$ odpovídajících různým množinám *I*, tedy

$$\textstyle\sum_{i\in I}(E_i) \quad I = \varnothing \quad \Leftrightarrow \quad \mathbf{0}$$

$$\textstyle\sum_{i\in I}(E_i) \quad I = \{\,1\,\} \quad \Leftrightarrow \quad E_1$$

$$\textstyle\sum_{i\in I}(E_i) \quad I = \{\,1, 2, 3\,\} \quad \Leftrightarrow \quad (E_1) + (E_2) + (E_3)$$

Je zřejmé, že množina výrazů $\mathcal{Z}$ je uzavřena vzhledem k oběma operacím již z definice. Potom čtveřice ($\mathcal{Z}$, +, *Act*, ·) utváří triviální algebraickou strukturu a můžeme již nyní zapsat například rovnost

$$E = \alpha\cdot(F + \beta\cdot A) \qquad E, F \in \mathcal{Z} \quad \alpha, \beta \in Act \quad A \in K$$

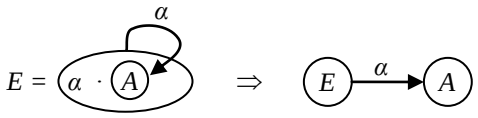
Máme tedy definován jazyk $\mathcal{Z}$, jehož syntaxe tvoří zmíněnou algebraickou strukturu. Avšak zde se jedná spíše o zajímavost, neboť tuto algebraickou strukturu v zájmu sémantiky vzápětí opustíme.

## 3.3 Sémantika

Až dosud nemají výrazy z množiny $\mathcal{Z}$ žádný význam. Ten jim poskytne teprve přechodový systém $(\mathcal{Z}, Act, \mathcal{R})$, jehož přechodová relace $\mathcal{R}$ (tedy podmnožina $\mathcal{Z} \times Act \times \mathcal{Z}$) je generována dvěma tzv. přechodovými pravidly

$$\begin{aligned} \alpha \in Act \ \wedge \ E \in \mathcal{Z} \quad &\Leftrightarrow \quad (\ \alpha{\cdot}(E),\ \alpha,\ E\ ) \ \in \ \mathcal{R} \\ j \in I \ \wedge \ (E_j, \alpha, F) \in \mathcal{R} \quad &\Leftrightarrow \quad (\ \textstyle\sum_{i\in I}(E_i),\ \alpha,\ F\ ) \ \in \ \mathcal{R} \end{aligned} \tag{8}$$

přičemž například první pravidlo znamená, že výraz $\alpha{\cdot}(E)$ se promění na $E$ právě provedením akce $\alpha$ (tzn. účastí systému na onom handshake se svým okolím). Stručně řečeno, každý výraz z množiny $\mathcal{Z}$ je „ovinut" odpovídajícím fragmentem sémantického přechodového systému $(\mathcal{Z}, Act, \mathcal{R})$, jak ukazuje jednoduchý příklad

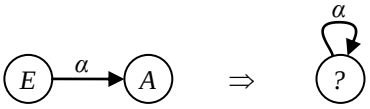



(9)

čímž je výrazům dodáván význam. Výraz tedy představuje určitý stav modelovaného systému i přípustný vývoj z tohoto stavu. Nicméně stojí alespoň za povšimnutí, že sémantický přechodový systém (resp. jeho přechodový graf) neobsahuje žádné cykly.

Avšak i po zavedení sémantického přechodového systému nemají konstanty (např. $A$) stále žádný význam. Navíc, výrazy stále představují především syntaktické útvary, mezi kterými neexistují žádné jiné vztahy, než ony dvě výše definovaná přechodová pravidla (8). Pro ilustraci použijeme přechodový systém daný přechodovým grafem


který zkusíme vyjádřit naším (až doposud vybudovaným) kalkulem. Je zřejmé, že bychom rádi uzavřeli (do smyčky) přechod z předchozího příkladu (9), tj. rádi bychom




avšak dosud nevíme, zda výrazy $E$ a $A$ jsou ekvivalentní či nikoliv. Přesto zapišme nabízející se rovnost

$$A = \alpha{\cdot}A \qquad\qquad \alpha \in Act \quad A \in K$$

přičemž je zřejmé, že tato rovnost není platná ve smyslu výše zmiňované algebraické struktury (vždyť $A$ a $\alpha{\cdot}A$ jsou různými prvky množiny $\mathcal{Z}$). Problém je, že tuto rovnost dosud nelze považovat za správnou z hlediska významu, neboť pro to nenalezneme oporu ve výše zavedených přechodových pravidlech (8). Díky nim sice víme, že se $\alpha{\cdot}A$ promění na $A$, ale již

nevíme zda $\alpha \cdot A$ je ekvivalentní s $A$, protože dosud význam konstanty $A$ neznáme. Jinými slovy, nevíme zda se $\alpha \cdot A$ může proměnit opět na $\alpha \cdot A$. Milner [1] si tedy trochu „vypomohl“ zavedením další, tzv. definiční rovnosti

$$A \stackrel{\text{def}}{=} \alpha \cdot A$$

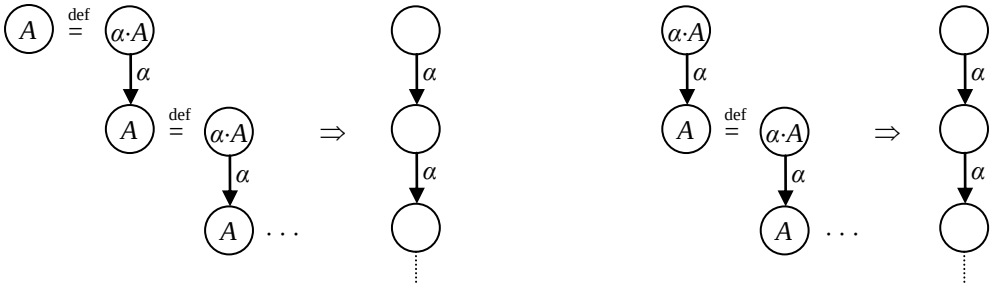
která dodává význam konstantě $A$ (a zároveň umožňuje překlenout absenci cyklů ve výše zavedeném sémantickém přechodovém systému).

Avšak také u standardní rovnosti je nezbytné si nějak „vypomoci“, neboť z hlediska významu je příliš striktní. Posuzuje totiž, přirozeně, zda obě její strany představují stejný prvek z množiny $\mathcal{3}$. Potom ji však mohou splnit jen syntakticky totožné výrazy, takže ačkoliv definiční rovnost zajistila (z hlediska významu) ekvivalenci výrazů $A$ a $\alpha \cdot A$, tak rovnost

$$A = \alpha \cdot A \tag{10}$$

splněna není. Milner [1] doporučuje chápat rovnost jako isomorfnost tzv. derivačních stromů, čímž však již definitivně opouštíme výchozí algebraickou strukturu. Derivační strom přísluší vždy konkrétnímu výrazu a je utvářen z odpovídajících fragmentů sémantického přechodového systému, které se postupně spojují dle definičních rovností (aby se dodržela struktura stromu, tak jsou u připojovaných fragmentů duplikovány sdílené úseky jednotlivých sledů). Ukážeme to na právě diskutované rovnosti (10). Výrazům $A$ a $\alpha \cdot A$ odpovídají fragmenty


ke kterým podle definiční rovnosti $A \stackrel{\text{def}}{=} \alpha \cdot A$ budeme postupně připojovat fragment odpovídající výrazu $\alpha \cdot A$ (tj. výrazu z pravé strany definiční rovnosti). Tímto pomyslným postupem, znázorněným na obrázku




získáme dva derivační stromy (v tomto případě nekonečné a degenerované). Milner uvádí, že „význam výrazu by měl být vlastností příslušného derivačního stromu při odhlédnutí od výrazů, které leží v uzlech; z tohoto hlediska derivační strom vyjadřuje vše co o výrazu potřebujeme vědět“ [1]. K tomu nyní jen poznamenejme, že derivační stromy bohužel vyjadřují dokonce více než potřebujeme vědět. Jde o stejný problém, na který jsme narazili při zkoumání struktury přechodů (viz závěr oddílu 2.4). Avšak zpět k diskutované rovnosti (10). Je zřejmé, že oba derivační stromy (bez výrazů v uzlech) jsou isomorfní a rovnost

$$A = \alpha \cdot A$$

chápaná jako isomorfnost derivačních stromů je již splněna. Souhrnně řečeno, právě ve formě derivačních stromů se spojuje příspěvek jednak sémantického přechodového systému a jednak definičních rovností, a to do jednotného a úplného vyjádření významu výrazů.

## 3.4 Vlastnosti

Jak jsme při zavádění sémantiky již zmínili, Milnerův základní kalkulus není algebraickou strukturou. Důvodem je především vzniklý rozdíl mezi ekvivalencí a totožností prvků nosné množiny $\mathcal{Z}$ (abychom zůstali na poli algebry, tak bychom s ohledem na požadovanou ekvivalenci museli sestrojit jinou nosnou množinu – například rozklad výchozí množiny $\mathcal{Z}$). Jinými slovy, výrazy kalkulu nemají „hodnoty“, rozuměno reprezentanty, které by představovaly význam nezávisle na způsobech jeho zápisu. Jistě lze namítnout, že takovými reprezentanty by mohly být právě derivační stromy. Bohužel derivační strom nelze vhodným způsobem formálně vyjádřit, neboť jeho vrcholy samotné musí být považovány za vzájemně nerozlišitelné a pouze pomocí symbolů akcí nad jeho hranami nelze popsat strukturu stromu. Proto lze s derivačními stromy také jen obtížně provádět nějaké formální operace (těžko si lze například představit, jak formálně vyjádřit rozklad množiny $\mathcal{Z}$ podle ekvivalence příslušných derivačních stromů). V důsledku toho derivační stromy poskytují pouze omezenou podporu (spočívající v jejich kreslení) pro formální usuzování nad samotným kalkulem.

Přes výše diskutované omezení je nutno říci, že zvolený způsob zavedení sémantiky a potažmo derivační stromy dostačují na posouzení vlastností operací kalkulu. Lze tedy ukázat, že operace + má následující vlastnosti (důkazy spočívají v řadě kreseb vždy prvního fragmentu zobecněných derivačních stromů a užití druhého přechodového pravidla):

- asociativita $E + (F + G) = (E + F) + G$ $E, F, G \in \mathcal{Z}$
- neutrální prvek $E + \mathbf{0} = E$
- komutativita $E + F = F + E$
- idempotence $E + E = E$

Operace · naopak (kromě uzavřenosti) nemá žádné vlastnosti. Kvůli redukci počtu závorek v zápisech výrazů je však dobré zdůraznit skutečnost vyplývající již ze syntaxe jazyka:

- asociativita zprava $\beta\cdot(\alpha\cdot E) = \beta\cdot\alpha\cdot E$ $\alpha, \beta \in Act \quad E \in \mathcal{Z}$

V tuto chvíli je naše rekonstrukce dovršena, neboť základní kalkulus v rámci námi zvoleného výřezu je již kompletní.

# 4 Navržený algebraický přístup

## 4.1 Popis vnějšího projevu

V rámci Milnerova přístupu je vnější projev popisován rovnou pomocí výrazů budovaného kalkulu. Milner totiž (jak jsme si ukázali v oddílu 3.3) bezprostředně navázal topologii přechodového grafu na vhodně zvolenou syntaxi těchto výrazů. On sám vysvětluje, že „definice přechodové relace sleduje strukturu výrazů, a tím jim dává význam“ [1]. Přestože je tento přístup na první pohled elegantní, tak přináší řadu podstatných problémů. Vzpomeňme na nutnost zavedení definiční rovnosti, dále na zástupné chápání (standardní) rovnosti výrazů jako isomorfismu jejich derivačních stromů a především na skutečnost, že spolu s topologií přechodového grafu se na výrazy přirozeně naváže více informace o struktuře přechodového systému, než je pro popis jeho vnějšího projevu nezbytné (jak jsme zjistili již při studiu struktury přechodů, viz závěr oddílu 2.4). Problém s nadbytečnou informací navázanou na výrazy kalkulu (v důsledku tedy i na jejich derivační stromy) se však projeví až při podrobnějším studiu ekvivalence (viz kapitola 5).

Vzhledem ke zmíněným problémům zde zvolíme odlišný přístup. Pro vybudování zamýšlené algebraické struktury tedy nevyužijeme bezprostředně přechodový systém, ale najdeme pro tento účel vhodného prostředníka. Přesněji řečeno, najdeme algebraicky lépe uchopitelný (formální) popis vnějšího projevu, který navíc bude oproštěn od nadbytečné informace nesené strukturou přechodového systému. Teprve nad těmito popisy vybudujeme algebraickou strukturu.

Požadovaný popis vnějšího projevu budeme pochopitelně odvozovat z přechodového systému. Proto si stručně připomeňme jeho definici (z oddílu 2.1) s tím, že podmínky přechodů jsou zde ztotožněny s akcemi. Přechodový systém je tedy trojice

$$(S, \Lambda, R^{\Lambda})$$

kde $S$ je konečná množina stavů, $\Lambda$ je konečná množina akcí a $R^{\Lambda}$ je přechodová relace. Přesněji řečeno, $R^{\Lambda} \subseteq S \times \Lambda \times S$, přičemž platí

$$(A, \alpha, B) \in R^{\Lambda} \qquad\qquad A, B \in S \quad \alpha \in \Lambda$$

právě když existuje přechod ze stavu $A$ do stavu $B$ označený akcí $\alpha$. Každý přechod přitom představuje nějaký vnitřní děj modelovaného systému. Část tohoto děje je vždy exponována do okolí modelovaného systému a právě tato část je reprezentována příslušnou akcí. Avšak akce není míněna jako pouhý (okolím modelovaného systému pozorovatelný) průvodní jev vnitřního děje. Na příslušné části vnitřního děje (reprezentovaného akcí) se okolí modelovaného systému musí podílet, tj. musí dojít k interakci s okolím. Tím akce taktéž naplňuje onu roli (nutné) podmínky přechodu. Nakonec si ještě připomeňme, že přechodový systém je vždy právě v jednom ze svých stavů a nemůže tedy nastat více přechodů najednou. Proto jakékoliv provedení (někdy nazývané také jako běh) přechodového systému má vždy formu posloupnosti přechodů.

**Přípustná provedení**

Přechodový systém obvykle umožňuje mnoho provedení (tj. posloupností přechodů), avšak pouze některé z nich dobře charakterizují, jak je modelovaný systém schopen interagovat se svým okolím. Proto se zaměříme spíše na taková provedení, jaká přechodový systém připustí, než jaká jsou možná. Přesněji řečeno, zavedeme (nejdříve jen intuitivně) pojem přípustné provedení. Pro objasnění vezměme například dva přechodové systémy


nacházející se ve stavu $A$. Oba přechodové systémy mají jedno nekonečné možné provedení (akce nad jeho přechody tvoří posloupnost $\alpha\alpha\alpha\ldots$) a nekonečně mnoho konečných možných provedení (akce nad jejich přechody tvoří posloupnosti $\alpha$, $\alpha\alpha$, $\alpha\alpha\alpha$, …). Znamená to, že oba mají stejný vnější projev? Nikoliv. Levý přechodový systém připustí právě to nekonečné provedení, což znamená, že modelovaný systém je trvale schopen provádět akci $\alpha$. Naproti tomu pravý přechodový systém připustí pouze ta konečná provedení, což znamená, že modelovaný systém je schopen provést nějaký (navíc nepředpověditelný) počet akcí $\alpha$, ale není ji schopen provádět trvale.

Nyní již přistoupíme k odpovídajícímu formálnímu vyjádření. Přípustné provedení definujeme jako určitý sled v přechodovém grafu, resp. jako posloupnost přechodů

$$(\ (X_i, \alpha_i, X_{i+1})\ )_{i \in I} \qquad X_1 = A \quad (X_i, \alpha_i, X_{i+1}) \in R^{\Lambda}$$

kde $R^{\Lambda}$ je přechodová relace, $A$ je počáteční stav provedení; přičemž $I$ je neprázdná množina indexů, která je

- konečná, tj. $I = \{ 1, 2, \ldots, n \}$, právě když stav $X_{n+1}$ je koncovým stavem (resp. stavem, ze kterého v přechodovém grafu nevystupuje žádný přechod), anebo
- nekonečná, tj. $I = \{ 1, 2, \ldots \}$, právě když existuje index $i$ takový, že v přechodovém grafu není ze stavu $X_i$ dosažitelný žádný koncový stav.

Pro danou přechodovou relaci $R^{\Lambda}$ a daný počáteční stav $A$ pak označíme množinu všech přípustných provedení jako

$$\epsilon^{\Lambda}{}_{A}$$

přičemž si povšimněme, že množina $\epsilon^{\Lambda}{}_{A}$ je prázdná, právě když počáteční stav $A$ je zároveň koncovým stavem (vzhledem k požadavku na neprázdnost množiny indexů $I$ totiž neexistují ani prázdná provedení).

**Interpretace akcí**

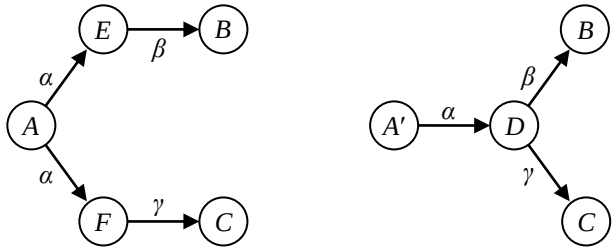
Výše zavedená množina $\epsilon^{\Lambda}{}_{A}$ již popisuje schopnost modelovaného systému, nacházejícího se ve stavu $A$, interagovat se svým okolím. Avšak my potřebujeme odpovídající popis pouze vnějšího projevu (tj. pouze z hlediska okolí modelovaného systému) a protože množina $\epsilon^{\Lambda}{}_{A}$

zachycuje i strukturu přechodového systému, tak popisuje dokonce víc než potřebujeme (jak jsme zjistili již při studiu struktury přechodů, viz závěr oddílu 2.4). Přesněji řečeno, rádi bychom odhlédli od vnitřních dějů v modelovaném systému. Zkusme tedy získat žádaný popis odvozením množiny všech přípustných posloupností akcí, tj. definováním množiny

$$\Pi^{\Lambda}{}_{A} = \{ \ ( \alpha_i )_{i \in I} : ( \ (X_i, \alpha_i, X_{i+1}) \ )_{i \in I} \in \mathcal{C}^{\Lambda}{}_{A} \ \}$$

která obsahuje takové posloupnosti akcí, jaké vyplývají ze všech provedení, které přechodový systém ve stavu $A$ připouští.

Všimněme si, že právě jsme vlastně jen formálně definovali množinu $\Pi$, kterou jsme si intuitivně uvědomili již při prvotním studiu přechodových systémů (viz oddíl 2.3). Odtud však také navíc víme, že kvůli nedeterminismu přechodových systémů tato množina není dostatečně výstižným popisem vnějšího projevu. Například dva následující přechodové systémy




(11)

mají totožnou množinu všech přípustných posloupností akcí, tj. platí

$$\Pi^{\Lambda}{}_{A} = \Pi^{\Lambda}{}_{A'} = \{ \alpha\beta, \alpha\gamma \}$$

což však není správně. Vždyť pravý přechodový systém provedení posloupnosti $\alpha\beta$ (a analogicky $\alpha\gamma$) připustí vždy, avšak levý přechodový systém ji připustit vždy nemusí. Přesněji řečeno, provedení akce $\alpha$ může vyvolat přechod do stavu $F$, v němž přechodový systém nepřipustí následné provedení akce $\beta$. Nicméně množiny $\Pi^{\Lambda}{}_{A}$ a $\Pi^{\Lambda}{}_{A'}$ tento rozdíl nezachycují.

O nedostatečné výstižnosti množiny $\Pi$ jsme již věděli, nyní se však hlouběji zamyslíme nad tím, co se v popsaném příkladu (11) vlastně stalo z hlediska modelovaného systému. Modelovaný systém v určitém stavu, abstrahovaném jako $A$, je nepochybně schopen se podílet na interakci, prostřednictvím akce $\alpha$, se svým okolím. Přesněji řečeno, v modelovaném systému mohou nastat nějaké dva vnitřní děje, abstrahované jako přechody do $E$ a $F$, jejichž exponovaná část je stejná, tj. z obou vnitřních dějů je exponována stejná akce $\alpha$ do okolí. Takže oba tyto vnitřní děje jsou nerozlišitelné z hlediska interakce samotné. Záleží pouze na modelovaném systému, který z obou vnitřních dějů je vyvolán nadcházející interakcí (všimněme si, že přechodový systém tento aspekt modelovaného systému nijak nevyjadřuje, resp. pouze jej rámcově zachycuje svým nedeterminismem). Nicméně oba tyto vnitřní děje jsou rozdílné a především mají rozdílné důsledky, takže od nich nemůžeme zcela odhlédnout (jako u automatů, popř. deterministických přechodových systémů). Podstatné přitom je, že jedna akce může mít různé důsledky také z vnějšího pohledu. Jinými slovy, akce je modelovaným systémem nějak interpretována, a právě to musíme vzít v úvahu. Takže zavedeme interpretaci akce (stručně jen interpretaci) jako dvojici $(\alpha, \Phi)$ zapisovanou pro přehlednost jako

$$\alpha^{\Phi} \qquad\qquad \alpha \in \Lambda \quad \Phi \subseteq \Lambda$$

přičemž interpretace $\alpha^{\Phi}$ představuje akci $\alpha$ společně s jejím bezprostředním důsledkem. Navíc definujeme také množinu všech interpretací, tj. množinu

$$\Psi = \{\ \alpha^{\Phi} : \alpha \in \Lambda,\ \Phi \subseteq \Lambda\ \}$$

Pak ke každému přechodovému systému $(S, \Lambda, R^{\Lambda})$, kterým jsme původně modelovali nějaký systém, můžeme sestrojit rozšířený přechodový systém definovaný trojicí

$$(S, \Psi, R^{\Psi})$$

kde $S$ je táž množina stavů jako v původním přechodovém systému $(S, \Lambda, R^{\Lambda})$, $\Psi$ je konečná množina interpretací a $R^{\Psi}$ je přechodová relace. Přesněji, $R^{\Psi} \subseteq S \times \Psi \times S$, přičemž platí

$$(A, \alpha^{\Phi(B)}, B) \in R^{\Psi} \quad \Leftrightarrow \quad (A, \alpha, B) \in R^{\Lambda}$$

s tím, že $\Phi(X)$ je množina akcí všech přechodů vystupujících ze stavu $X$, která je definovaná následovně

$$\Phi(X) = \{\ \beta : (X, \beta, Y) \in R^{\Lambda}\ \}$$

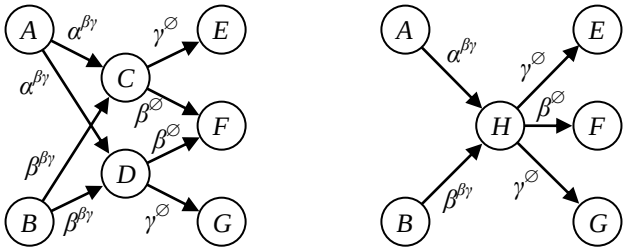
Je zřejmé, že rozšířený přechodový systém (sestrojený popsaným způsobem) plně reprezentuje i ten původní, tj. platí

$$(S, \Psi, R^{\Psi}) \quad \Leftrightarrow \quad (S, \Lambda, R^{\Lambda})$$

protože jsme pouze přidali nějaké redundantní (avšak užitečné) informace. Potom ani není nijak překvapující, že rozšířený přechodový systém také plně zachovává význam.

**Odkládaný nedeterminismus**

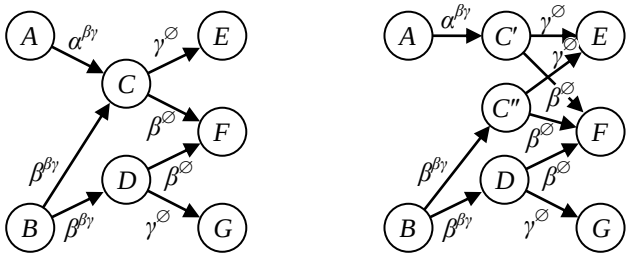
Ačkoliv jsme v rámci rozšířeného přechodového systému doplnili do označení nad přechody informaci o důsledku akcí, tzn. akce samotné jsme nahradili jejich interpretací, tak stále ještě není jisté, zda to postačí k získání dostatečně výstižného popisu vnějšího projevu. Důvodem je, že zavedené interpretace zahrnují informaci pouze o bezprostředním důsledku příslušné akce. Nejistota tedy tkví v tom, zda tato informace je již dostačující. Vždyť obecně je rozšířený přechodový systém stále nedeterministický i ve smyslu interpretací. Prozkoumejme proto následující příklad




ve kterém rozšířený přechodový systém vlevo je nedeterministický pro interpretace $\alpha^{\beta\gamma}$ a $\beta^{\beta\gamma}$ v odpovídajících stavech $A$ a $B$ (všimněme si také, že zapisujeme všechny interpretace, např. $\alpha^{\{\beta,\gamma\}}$, pro přehlednost ve tvaru $\alpha^{\beta\gamma}$). Nicméně stavy $C$ a $D$ nejsou z vnějšího pohledu rozlišitelné. Vždyť oba tyto stavy jsou dosažitelné stejnými akcemi $(\alpha, \beta)$ ze stejných stavů $(A, B)$ a především pak v obou těchto stavech jsou bezprostředně přípustné stejné akce $(\beta, \gamma)$. Takže

oba tyto stavy můžeme sloučit, díky čemuž získáme upravený přechodový systém vyobrazený vpravo. Všimněme si, že jsme tím vlastně jen odložili nedeterminismus ze stavů $A$ a $B$ do sloučeného stavu $H$.

Předchozí příklad popisuje situaci, kde je pro každý vyšetřovaný stav $Y_i$ (např. $C$ a $D$) splněna následující podmínka: jestliže existuje nějaký přechod $(X, \alpha^{\Phi}, Y_i)$, potom jistě existuje také přechod $(X, \alpha^{\Phi}, Y_j)$ pro libovolné $j \neq i$ (tzn. ze stejného stavu jsou stejnou interpretací dosažitelné všechny vyšetřované stavy). Následující příklad již popisuje situaci, kde tato podmínka splněna není. Mějme tedy rozšířený přechodový systém




vyobrazený vlevo. Je zřejmé, že stav $C$ nesplňuje zmiňovanou podmínku, takže jej rozdělíme na stavy $C'$ a $C''$, čímž získáme upravený přechodový systém vyobrazený vpravo. Potom již můžeme sloučit stavy $C''$ a $D$ (stejně jako v předchozím příkladu). Tím se nám opět podařilo odložit nedeterminismus ze stavu $B$.

Zde je ovšem třeba podotknout, že počáteční stav (resp. jakýkoliv stav, do kterého je možné modelovaný systém „nastavit" jinak než interakcemi s okolím) vyžaduje speciální péči. Počáteční stav, např. $C$, musí být zpracován, jako kdyby existoval speciální přechod $(\partial, \partial^{\Phi(C)}, C)$, přičemž $\partial$ a $\partial$ nejsou užity žádnými jinými přechody. S tímto předpokladem smí být výše popsané rozdělování a slučování stavů použito také na počáteční stav.

Uvedeným postupem můžeme v rozšířeném přechodovém systému nedeterminismus postupně odkládat tak dlouho, dokud není ve smyslu interpretací eliminován (ve smyslu akcí samozřejmě eliminovat nelze). Výsledný přechodový systém pak nazveme vnějším přechodovým systémem, který definujeme jako trojici

$$(Q, \Psi, L^{\Psi})$$

kde $Q$ je konečná množina stavů (tj. množina $S$ po všech úpravách, resp. po veškerém rozdělování a slučování stavů), $\Psi$ je konečná množina interpretací a $L^{\Psi}$ je přechodová relace (tj. relace $R^{\Psi}$ po všech úpravách). Přitom vztah mezi rozšířeným přechodovým systémem a příslušným vnějším přechodovým systémem je již pouze implikace, tj.

$$(S, \Psi, R^{\Psi}) \quad \Rightarrow \quad (Q, \Psi, L^{\Psi})$$

protože (obecně) ztrácíme informaci o vnitřních dějích. Nicméně příslušné vnější a rozšířené přechodové systémy nelze rozlišit z vnějšího pohledu (resp. z hlediska okolí modelovaného systému) a vnější přechodové systémy jsou ve smyslu interpretací již deterministické.

**Schopnost a stav**

Právě skutečnost, že vnější přechodový systém je ve smyslu interpretací již deterministický, nám (pro modelovaný systém v daném počátečním stavu) dovoluje získat výstižný popis jen vnějšího projevu. Díky determinismu totiž posloupnosti interpretací jednoznačně určují příslušná provedení (resp. sledy v přechodovém grafu). Takže tyto posloupnosti nám umožní plně popsat vnější projev, aniž bychom zachycovali nadbytečnou informaci o struktuře příslušného přechodového systému.

Nejdříve pro danou přechodovou relaci $L^{\Psi}$ a daný počáteční stav $A$ označíme množinu všech přípustných provedení jako

$$\in^{\Psi}_{A}$$

analogicky k výše zavedené $\in^{\Lambda}_{A}$. Pomocí množiny $\in^{\Psi}_{A}$ potom pro příslušný vnější přechodový systém $(Q, \Psi, L^{\Psi})$ a daný počáteční stav $A \in Q$ odvodíme množinu všech přípustných posloupností interpretací, tj. definujeme množinu

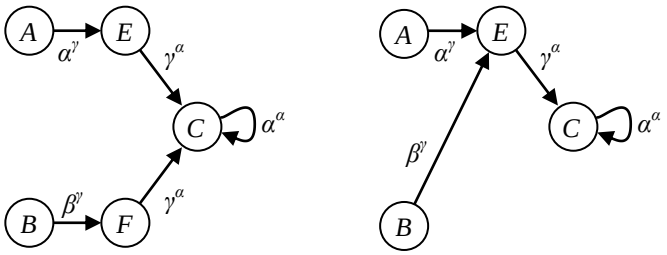
$$\Pi^{\Psi}_{A} = \{ ( \alpha_i^{\Phi(Xi+1)} )_{i \in I} : ( (X_i, \alpha_i^{\Phi(Xi+1)}, X_{i+1}) )_{i \in I} \in \in^{\Psi}_{A} \}$$

která obsahuje takové posloupnosti interpretací, jaké vyplývají ze všech provedení, které vnější přechodový systém ve stavu $A$ připouští. Protože tato množina je oním požadovaným popisem vnějšího projevu, tak ji nazveme jako *schopnost* (neboť vystihuje, jak je modelovaný systém v daném počátečním stavu schopen interagovat se svým okolím).

Dále prozkoumáme vztah mezi stavy vnějšího přechodového systému a *schopnostmi*. Je zcela zřejmé, že platí

$$A \Rightarrow \Pi^{\Psi}_{A}$$

což znamená, že stav $A$ určuje *schopnost* $\Pi^{\Psi}_{A}$ v rámci konkrétního vnějšího přechodového systému. Potom však stavy určující stejnou *schopnost*, můžeme označit za ekvivalentní. Pro ilustraci využijme následující příklad, kde vnější přechodový systém




vyobrazený vlevo má dva ekvivalentní stavy (tj. stavy $E$ a $F$), protože jimi určené *schopnosti* jsou následující

$$\Pi^{\Psi}_{E} = \Pi^{\Psi}_{F} = \{ \gamma^{\alpha}\alpha^{\alpha}\alpha^{\alpha}\alpha^{\alpha}\alpha^{\alpha}\ldots \}$$

což znamená, že vnější přechodový systém smí v obou případech provést jen a pouze ty stejné akce se stejnými důsledky. Potom nezáleží, ve kterém z obou stavů se tento přechodový systém nachází. Takže můžeme tyto stavy sloučit, aniž by to mělo nějaký dopad na jakoukoliv *schopnost* (všimněme si, že se jedná o podobný postup jako u automatů, viz Hopcroft a

spol. [7]), čímž dostaneme redukovaný vnější přechodový systém vyobrazený vpravo. Když jsou sloučeny veškeré ekvivalentní stavy, tak nemohou existovat žádné dva různé stavy určující stejnou *schopnost*, tj. platí

$$A \Leftrightarrow \Pi^{\Psi}{}_{A}$$

což znamená, že *schopnost* dokonce reprezentuje stav redukovaného vnějšího přechodového systému (resp. reprezentuje stav rozlišitelný okolím modelovaného systému).

Závěrem ještě stručně srovnejme *schopnosti*, resp. množiny $\Pi^{\Psi}$, se dříve diskutovanými přístupy. Oproti množinám $\Pi^{\Lambda}$ (popř. *completed traces*) zachycují i nedeterminismus. Přesněji řečeno, zachycují pouze takové projevy nedeterminismu, které jsou z vnějšího pohledu (tzn. okolím modelovaného systému) rozpoznatelné. Na druhou stranu, oproti Milnerovým derivačním stromům (viz oddíl 3.3) však nezachycují nadbytečnou informaci o vnitřní struktuře (tzn. o topologii přechodového grafu), na které vnější projev nezávisí.

Pokud tedy okolí smí se systémem, modelovaným pomocí přechodového systému, manipulovat pouze interakcemi prostřednictvím akcí, tak se právě *schopnosti* jeví jako vhodný popis vnějšího projevu tohoto systému.

## 4.2 Algebraická struktura

Popis vnějšího projevu tedy k dispozici máme. Přesněji řečeno, ke každému přechodovému systému v libovolném stavu $X$ nyní umíme přiřadit výše definovanou schopnost $\Pi^{\Psi}{}_{X}$. A nad těmito schopnostmi již vybudujeme (složenou) algebraickou strukturu.

**Nosné množiny**

Nejdříve je potřeba zavést nosné množiny, kterými v našem případě budou množina akcí a množina schopností. Množina akcí $\Lambda$ je již dána a množinu schopností definujeme jako

$$K = 2^{\Psi^{\infty}} \qquad\qquad \Psi^{\infty} = \Psi^{*} \cup \Psi^{\omega}$$

což znamená, že $K$ je (nekonečná) množina všech podmnožin množiny $\Psi^{\infty}$, která je množinou všech (konečných a nekonečných) posloupností interpretací. Uvědomme si přitom, že každá schopnost $\Pi^{\Psi}{}_{X}$ je podmnožinou množiny $\Psi^{\infty}$, takže můžeme psát

$$\Pi^{\Psi}{}_{X} \in K$$

a všimněme si také, že množina $K$ je v důsledku určena pouze množinou akcí $\Lambda$. To znamená, že množina $K$ neobsahuje jen schopnosti určitého přechodového systému, ale představuje universum schopností k dané množině akcí $\Lambda$ (jak ostatně od nosné množiny očekáváme).

Dále zavedeme proměnné, které mohou nabývat hodnot právě z množiny $K$. Tyto proměnné budeme zapisovat tučnými písmeny (tj. $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{C}$, …) a jejich význam definujeme jako

$$\boldsymbol{A} = \Pi^{\Psi}{}_{A}$$

což znamená, že proměnná $\boldsymbol{A}$ reprezentuje jak stav $A$, tak schopnost přechodového systému v tomto stavu.

**Operace a vlastnosti**

Nyní přikročíme k definici operací se zamýšlenými vlastnostmi. Nejdříve definujeme jednoduchou strukturu $(K, +)$ jakožto komutativní idempotentní monoid. Takže operace + musí být funkcí $K \times K \to K$, která splňuje axiomy uzavřenosti, asociativity, neutrálního prvku, komutativity a idempotence. V souladu s tím definujeme + jako sjednocení dvou schopností, takže například rovnost

$$\boldsymbol{C} = \boldsymbol{A} + \boldsymbol{B} \qquad\qquad \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in K$$

můžeme zapsat jako

$$\Pi^{\Psi_C} = \Pi^{\Psi_A} \cup \Pi^{\Psi_B}$$

přičemž je velmi snadné ukázat, že takto definovaná operace + splňuje všechny výše zmíněné axiomy:

- Axiom uzavřenosti – Obě množiny $\Pi^{\Psi_A}$ a $\Pi^{\Psi_B}$ jsou podmnožinami množiny $\Psi^{\infty}$. Přitom nosná množina $K$ je potenční množinou množiny $\Psi^{\infty}$, tj. $K$ obsahuje všechny podmnožiny množiny $\Psi^{\infty}$. A protože sjednocením podmnožin dostaneme opět podmnožinu, tak $K$ obsahuje také množinu $\Pi^{\Psi_C}$. Takže pro jakékoliv dvě schopnosti $\boldsymbol{A}$ a $\boldsymbol{B}$ platí, že $\boldsymbol{A} + \boldsymbol{B}$ je opět schopnost, resp. je opět z nosné množiny $K$.

- Axiom neutrálního prvku – Nosná množina $K$ (jakožto potenční množina) obsahuje také prázdnou množinu. Přitom tato prázdná množina přirozeně vyjadřuje „žádnou" schopnost (resp. neschopnost modelovaného systému v příslušném stavu jakkoliv interagovat se svým okolím). Vzhledem k definici operace + můžeme tedy snadno zavést neutrální prvek (zapisovaný tučnou nulou) definovaný jako

  $$\mathbf{0} = \varnothing$$

  protože sjednocením libovolné množiny s prázdnou množinou dostaneme opět původní množinu. Takže rovnost $\boldsymbol{A} + \mathbf{0} = \mathbf{0} + \boldsymbol{A} = \boldsymbol{A}$ platí pro jakoukoliv schopnost $\boldsymbol{A}$.

- Axiomy asociativity, komutativity a idempotence – Vzhledem k definici operace + jsou tyto axiomy splněny samozřejmě, protože sjednocení množin je asociativní, komutativní i idempotentní. Takže příslušné rovnosti $(\boldsymbol{A} + \boldsymbol{B}) + \boldsymbol{C} = \boldsymbol{A} + (\boldsymbol{B} + \boldsymbol{C})$, $\boldsymbol{A} + \boldsymbol{B} = \boldsymbol{B} + \boldsymbol{A}$ a také $\boldsymbol{A} + \boldsymbol{A} = \boldsymbol{A}$ platí pro jakékoliv schopnosti $\boldsymbol{A}$, $\boldsymbol{B}$ a $\boldsymbol{C}$.

Máme tedy jednoduchou algebraickou strukturu $(K, +)$, jejíž operace má stejné vlastnosti jako stejnojmenná operace v Milnerově kalkulu (viz oddíl 3.4). Následně již definujeme složenou strukturu $(K, +, \Lambda, \cdot)$ jakožto rozšíření struktury $(K, +)$ tak, že operace $\cdot$ je funkcí $\Lambda \times K \to K$, která splňuje axiom uzavřenosti. V souladu s tím definujeme $\cdot$ jako zřetězení interpretace a schopnosti, takže například rovnost

$$\boldsymbol{B} = \alpha \cdot \boldsymbol{A} \qquad\qquad \boldsymbol{A}, \boldsymbol{B} \in K \quad \alpha \in \Lambda$$

můžeme zapsat jako

$$\Pi^{\Psi_B} = \{\ \alpha^{\Phi(A)}\zeta\ :\ \zeta \in \Pi^{\Psi_A}\ \} \qquad \Pi^{\Psi_A} \neq \emptyset$$
$$\Phi(A) = \{\ \lambda : \lambda^{\Phi}\xi \in \Pi^{\Psi_A}\ \}$$

nebo

$$\Pi^{\Psi_B} = \{\ \alpha^{\emptyset}\ \} \qquad \Pi^{\Psi_A} = \emptyset$$

přičemž opět velmi snadno nahlédneme, že takto definovaná operace · splňuje všechny požadované vlastnosti:

- Axiom uzavřenosti – Množina $\Pi^{\Psi_A}$ je podmnožinou množiny $\Psi^{\infty}$. Přitom množina $\Psi^{\infty}$ obsahuje všechny posloupnosti interpretací. Jelikož zřetězením interpretace $\alpha^{\Phi(A)}$ a posloupnosti $\zeta$ dostaneme opět posloupnost interpretací, potom také $\Pi^{\Psi_B}$ je podmnožinou množiny $\Psi^{\infty}$. A protože nosná množina $K$ obsahuje všechny podmnožiny množiny $\Psi^{\infty}$, tak $K$ obsahuje také množinu $\Pi^{\Psi_B}$. Takže pro jakoukoliv akci $\alpha$ a schopnost $\boldsymbol{A}$ platí, že $\alpha{\cdot}\boldsymbol{A}$ je opět schopnost, resp. je opět z nosné množiny $K$.

- Asociativita zprava – Ačkoliv jednostranná asociativita není vlastností ve smyslu axiomu, tak zůstává podstatnou z hlediska jazyka zápisu výrazů. Asociativita zprava je zde totiž dána již signaturou funkce $\Lambda \times K \to K$. Jinými slovy, pořadí provádění operací je striktně předepsáno již samotným jazykem. Takže rovnost $\beta{\cdot}\alpha{\cdot}\boldsymbol{A} = \beta{\cdot}(\alpha{\cdot}\boldsymbol{A})$ platí pro jakékoliv akce $\alpha$, $\beta$ a schopnost $\boldsymbol{A}$.

V tuto chvíli je zamýšlená (složená) algebraická struktura již vybudována. Je zřejmé, že obě její operace mají stejné vlastnosti jako stejnojmenné operace v Milnerově kalkulu. Také jazyk zápisu samotných výrazů je s Milnerovým kalkulem shodný. Naopak rovnost je zde chápána standardně (tzn. jako totožnost prvků nosné množiny $K$ vyjadřovaných výrazy z obou stran rovnosti). Navíc zde není potřeba Milnerovy definiční rovnosti, neboť například rovnost

$$\boldsymbol{A} = \alpha{\cdot}\boldsymbol{A} \qquad \boldsymbol{A} \in K \quad \alpha \in \Lambda$$

lze standardně chápat jako rovnici, jejímž řešením je určité $\boldsymbol{A}$. Díky tomu (a díky vhodnému způsobu konstrukce množin $\Pi^{\Psi_X}$ a potažmo nosné množiny $K$) je však přepis přechodového grafu na soustavu rovnic stejně přímočarý jako u Milnerova kalkulu, což si ihned ukážeme na konkrétních příkladech.

**Příklady**

Návaznost přechodových systémů přes algebraické rovnice až na schopnosti (resp. množiny $\Pi^{\Psi_X}$, nad nimiž je algebraická struktura vybudována) si ukážeme jak na dvou rozhodujících příkladech, tak (pro ilustraci) i na jednom komplexnějším příkladu. Za prvé, systému daného přechodovým grafem


odpovídá jednoduchá algebraická rovnice

$$\boldsymbol{A} = \alpha \cdot \boldsymbol{A}$$

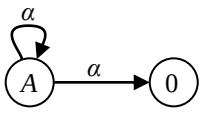
jejímž řešením je schopnost

$$\boldsymbol{A} \;=\; \{\, \alpha^{\alpha}\alpha^{\alpha}\alpha^{\alpha}\ldots \,\}$$

přičemž správnost tohoto řešení můžeme ověřit jeho dosazením do rovnice. Jakmile uvedenou schopnost $\boldsymbol{A}$ dosadíme do pravé strany rovnice, tak dostaneme výslednou schopnost

$$\alpha \cdot \boldsymbol{A} \;=\; \alpha \cdot \{\, \alpha^{\alpha}\alpha^{\alpha}\alpha^{\alpha}\ldots \,\} \;=\; \{\, \alpha^{\alpha}\alpha^{\alpha}\alpha^{\alpha}\ldots \,\}$$

která je totožná se schopností $\boldsymbol{A}$. Ovšem, vždyť nekonečná posloupnost $\alpha^{\alpha}\alpha^{\alpha}\alpha^{\alpha}\ldots$ rozšířená o prefix $\alpha^{\alpha}$ (tzn. o interpretaci akce $\alpha$) je stále nekonečnou posloupností $\alpha^{\alpha}\alpha^{\alpha}\alpha^{\alpha}\ldots$. Takže řešení rovnici skutečně vyhovuje a lze snadno nahlédnout, že je řešením jediným. Uvedená rovnice tedy (implicitně) vystihuje vnější projev daného systému stejně, jako jej (explicitně) vystihuje uvedené řešení. Za druhé, systému daného přechodovým grafem




již odpovídá odlišná algebraická rovnice

$$\boldsymbol{A} = \alpha \cdot \boldsymbol{A} + \alpha \cdot \boldsymbol{0}$$

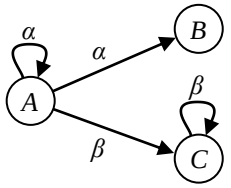
jejímž řešením je schopnost

$$\boldsymbol{A} \;=\; \{\, \alpha^{\varnothing},\, \alpha^{\alpha}\alpha^{\varnothing},\, \alpha^{\alpha}\alpha^{\alpha}\alpha^{\varnothing},\, \ldots \,\}$$

přičemž správnost tohoto řešení lze opět snadno ověřit jeho dosazením do rovnice. Skutečně, dosazením do pravé strany rovnice dostaneme výslednou schopnost

$$\begin{aligned} \alpha \cdot \boldsymbol{A} + \alpha \cdot \boldsymbol{0} \;&=\; \alpha \cdot \{\, \alpha^{\varnothing},\, \alpha^{\alpha}\alpha^{\varnothing},\, \alpha^{\alpha}\alpha^{\alpha}\alpha^{\varnothing},\, \ldots \,\} \;+\; \alpha \cdot \varnothing \;= \\ &=\; \{\, \alpha^{\alpha}\alpha^{\varnothing},\, \alpha^{\alpha}\alpha^{\alpha}\alpha^{\varnothing},\, \ldots \,\} \;+\; \{\, \alpha^{\varnothing} \,\} \;= \\ &=\; \{\, \alpha^{\varnothing},\, \alpha^{\alpha}\alpha^{\varnothing},\, \alpha^{\alpha}\alpha^{\alpha}\alpha^{\varnothing},\, \ldots \,\} \end{aligned}$$

která souhlasí s levou stranou rovnice. Nakonec si ještě stručně ukážeme komplexnější systém, resp. následující přechodový graf společně s odpovídající soustavou tří algebraických rovnic a jejím řešením




$$\begin{aligned} \boldsymbol{A} \;&=\; \alpha \cdot \boldsymbol{A} + \alpha \cdot \boldsymbol{B} + \beta \cdot \boldsymbol{C} \\ \boldsymbol{B} \;&=\; \boldsymbol{0} \\ \boldsymbol{C} \;&=\; \beta \cdot \boldsymbol{C} \end{aligned}$$

$$\begin{aligned}
\boldsymbol{A} &= \{\, \alpha^{\varnothing},\ \alpha^{\alpha\beta}\alpha^{\varnothing},\ \ldots,\ \beta^{\beta}\beta^{\beta}\ldots,\ \alpha^{\alpha\beta}\beta^{\beta}\beta^{\beta}\ldots,\ \alpha^{\alpha\beta}\alpha^{\alpha\beta}\beta^{\beta}\beta^{\beta}\ldots,\ \ldots \,\} \\
\boldsymbol{B} &= \varnothing \\
\boldsymbol{C} &= \{\, \beta^{\beta}\beta^{\beta}\ldots \,\}
\end{aligned}$$

přičemž správnost uvedeného řešení lze opět ověřit následujícím dosazením

$$\begin{aligned}
\alpha\cdot\boldsymbol{A} + \alpha\cdot\boldsymbol{B} + \beta\cdot\boldsymbol{C} &= \alpha\cdot\{\, \alpha^{\varnothing},\ \alpha^{\alpha\beta}\alpha^{\varnothing},\ \ldots,\ \beta^{\beta}\beta^{\beta}\ldots,\ \alpha^{\alpha\beta}\beta^{\beta}\beta^{\beta}\ldots,\ \alpha^{\alpha\beta}\alpha^{\alpha\beta}\beta^{\beta}\beta^{\beta}\ldots,\ \ldots \,\} + \\
&\quad + \alpha\cdot\varnothing + \beta\cdot\{\, \beta^{\beta}\beta^{\beta}\ldots \,\} = \\
&= \{\, \alpha^{\alpha\beta}\alpha^{\varnothing},\ \alpha^{\alpha\beta}\alpha^{\alpha\beta}\alpha^{\varnothing},\ \ldots,\ \alpha^{\alpha\beta}\beta^{\beta}\beta^{\beta}\ldots,\ \alpha^{\alpha\beta}\alpha^{\alpha\beta}\beta^{\beta}\beta^{\beta}\ldots,\ \ldots \,\} + \\
&\quad + \{\, \alpha^{\varnothing} \,\} + \{\, \beta^{\beta}\beta^{\beta}\ldots \,\} = \\
&= \{\, \alpha^{\varnothing},\ \alpha^{\alpha\beta}\alpha^{\varnothing},\ \ldots,\ \beta^{\beta}\beta^{\beta}\ldots,\ \alpha^{\alpha\beta}\beta^{\beta}\beta^{\beta}\ldots,\ \alpha^{\alpha\beta}\alpha^{\alpha\beta}\beta^{\beta}\beta^{\beta}\ldots,\ \ldots \,\} = \boldsymbol{A}
\end{aligned}$$

$$\boldsymbol{0} = \varnothing = \boldsymbol{B}$$

$$\beta\cdot\boldsymbol{C} = \beta\cdot\{\, \beta^{\beta}\beta^{\beta}\ldots \,\} = \{\, \beta^{\beta}\beta^{\beta}\ldots \,\} = \boldsymbol{C}$$

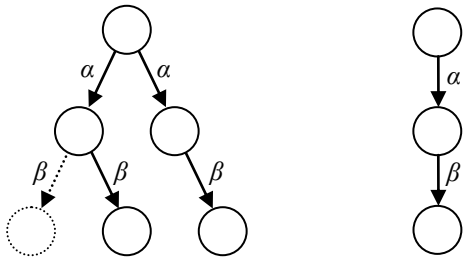
# 5 Ekvivalence systémů

## 5.1 Spornost ekvivalence

Výchozím pojetím ekvivalence výrazů v rámci Milnerova základního kalkulu, a tudíž i při jeho rekonstrukci (viz kapitola 3), je isomorfnost příslušných derivačních stromů. Při oné rekonstrukci jsme sice poznamenali, že derivační stromy nejsou nejvhodnějšími reprezentanty významu (ovšem pouze kvůli nemožnosti je vhodným způsobem formálně vyjádřit), ale zároveň jsme přisvědčili, že jimi lze prokázat deklarované vlastnosti (asociativita, idempotence apod.) operací základního kalkulu. Nicméně, nyní si ukážeme paradoxní příklad dvou ekvivalentních výrazů, jejichž derivační stromy ale nejsou isomorfní. Mějme dva výrazy

$$\alpha\cdot(\beta\cdot\mathbf{0} + \beta\cdot\mathbf{0}) + \alpha\cdot(\beta\cdot\mathbf{0}) \qquad\qquad \alpha\cdot(\beta\cdot\mathbf{0})$$

ke kterým sestrojíme příslušné derivační stromy (jelikož zde nefigurují definiční rovnosti, tak vlastně ve tvaru stromů pouze nakreslíme fragmenty sémantického přechodového systému odpovídající daným výrazům)




přičemž tečkovaný přechod pouze upozorňuje na nejednoznačný výklad původních Milnerových přechodových pravidel. V naší rekonstrukci základního kalkulu se totiž přechodová pravidla (8) zakládají na přechodové relaci $\mathit{Я} \subseteq \mathit{3} \times \mathit{Act} \times \mathit{3}$, tj. množině, ve které se přechod $(\ (\beta\cdot\mathbf{0})+(\beta\cdot\mathbf{0}),\ \beta,\ \mathbf{0}\ )$ nachází samozřejmě jen jednou, přestože vyplývá z obou podvýrazů $\beta\cdot\mathbf{0}$. V našem derivačním stromu se tečkovaný přechod tedy nevyskytuje. Oproti tomu Milnerova přechodová pravidla [1] mají tvar poněkud méně formálních přepisovacích pravidel, takže ani po zevrubném studiu jeho knihy není možné s jistotou usoudit, jakým způsobem by se v takovém případě použila. Proto v Milnerově derivačním stromu možná tečkovaný přechod figuruje. Přes zmíněnou nejistotu lze spolehlivě prohlásit, že výše sestrojené derivační stromy zjevně nejsou isomorfní. Takže derivačními stromy prokázat ekvivalenci daných výrazů nelze (právě zde se totiž projevuje již předeslané, že derivační stromy vyjadřují více než potřebujeme). Paradox příkladu potom spočívá ve skutečnosti, že ekvivalenci lze prokázat díky idempotenční vlastnosti (viz $E + E = E$) dokázané právě pomocí zde selhávajících derivačních stromů. Vzhledem k jednoduchosti příkladu lze snadno nahlédnout, že vnější projevy systémů odpovídajících daným výrazům by skutečně byly nerozlišitelné. Tento příklad tedy nezpochybňuje idempotenční vlastnost, ale definitivně vylučuje derivační stromy jako reprezentanty významu výrazů kalkulu.

Milner [1] později přichází s dalším druhem rovnosti, založeným na tzv. silné bisimulaci, která je již lépe formálně využitelná a jí indukovaná ekvivalence je kongruencí (tj. zachovává operace kalkulu), proto ji Milner označuje jako „první řádné pojetí ekvivalence“ [1]. Ostatně celou jeho knihou [1] se vine nit hledání, jakým způsobem ekvivalenci uchopit. A zde je dobré se na chvíli zastavit.

## 5.2 Co ekvivalencí systémů vlastně míníme?

Již v prvotním zkoumání samotných přechodových systémů (viz oddíl 2.3) jsme narazili na potřebu vyjasnit co ekvivalencí míníme. V tomto ohledu se Milner vztahuje k pojmu chování systému a uvádí, že „chování systému je přesně to, co je pozorovatelné a pozorovat systém znamená komunikovat s ním“ [1]. Obdobně jsme se také my vztahovali k pojmu vnější projev systému. Souhrnně lze říci, že společně považujeme dva systémy za ekvivalentní, právě a jen když nejsou vzájemně rozlišitelné z vnějšího pohledu. Snad nejpřesněji to lze vyjádřit tak, že „systémy nejsou ekvivalentní, právě a jen když existuje nějaký experiment, který systémy dokáže rozlišit“ [2]. Abychom si však ujasnili co míníme experimentem (a tedy i ekvivalencí) a přitom stále zůstali na dostatečně abstraktní úrovni, tak využijeme metaforu. Inspirujeme se článkem Blooma a Meyera [2] o experimentování s ekvivalencí a přistoupíme na poměrně přiléhavou *metaforu tlačítek*, která nám poskytne jakýsi etalon ekvivalence.

V rámci metafory budeme na systém nahlížet jako na černou skříňku s tlačítky označenými pomocí symbolů akcí z množiny $\Lambda$. Jedinou možností jak může experimentátor manipulovat s černou skříňkou, je postupné tisknutí jednotlivých tlačítek v libovolném pořadí (ve kterém se tlačítka mohou také libovolně opakovat). Ovšem jaká tlačítka na stisk budou reagovat a jaká nikoliv, již závisí na aktuálním stavu skříňky. Přitom reakce spočívá jednak ve zpětné odezvě experimentátorovi (tj. ten vždy pozná, že tlačítko na jeho stisk reagovalo) a jednak v případné změně aktuálního stavu skříňky (resp. pokud tlačítko na stisk nereagovalo, tak se stav nezměnil určitě). Samotné experimentování potom probíhá následovně. Experimentátor má k dispozici nekonečný počet stejných skříněk ve stejném (resp. počátečním) stavu. Postupně na každé skříňce pak provede právě jeden experiment spočívající v libovolné konečné posloupnosti stisků tlačítek. Ovšem nemůžeme čekat, až neúnavný experimentátor provede nekonečný počet experimentů a poskytne nám jejich záznam. Proto formálně vyjádříme raději předpovědi experimentů ve tvaru posloupnosti

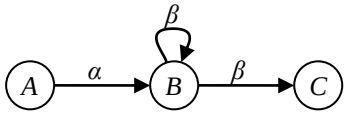
$$U_0\, \alpha_0\, U_1 \ldots \alpha_n\, U_{n+1} \qquad\qquad U_i \subseteq \Lambda \quad \alpha_i \in \Lambda$$

kde $U_i$ obsahuje symboly všech v daném stavu skříňky nereagujících tlačítek (tím předpovídáme vlastně celé posloupnosti stisků $\xi \in U_i^*$, při nichž se náš experimentátor žádné reakce nedočká) a následující $\alpha_i$ je symbolem jednoho z těch tlačítek, které v daném stavu skříňky reagují. Popsaná posloupnost tedy předpovídá určitou množinu experimentů, které náš experimentátor ve svém nekonečném úsilí dříve či později zaznamená.

Bloom a Meyer [2] diskutují několik variant *metafory tlačítek*. Zmiňme jednak metaforu prosvětlených tlačítek, kterou jsme se inspirovali my v předchozím odstavci a jednak metaforu tlačítek s duplikátorem, kterou oni zkoumali (v několika modifikacích) podrobněji. Posledně jmenovaná metafora se vyznačuje tím, že experimentátorovi umožňuje k černé skříňce kdykoliv (tzn. mezi jednotlivými stisky tlačítek) vytvořit libovolný počet duplikátů, na nichž lze provádět dílčí experimenty. Tato metafora dokáže systémy rozlišovat jemněji, avšak charak-

ter zde předpokládaných systémů neumožňuje přímé zjištění jejich stavu (mj. právě přímé zjištění stavu je principiálně nezbytnou součástí duplikace), proto v naší metafoře neumožňujeme vytvářet duplikáty černých skříněk. Stručně řečeno, experimentátor může s černými skříňkami manipulovat pouze tisknutím tlačítek, stejně jako okolí může se systémem interagovat pouze prostřednictvím akcí. Z jiného úhlu pohledu můžeme také říci, že umožňujeme duplikaci jen před prvním stiskem tlačítka, což bylo vyjádřeno jako dostupnost nekonečného počtu stejných skříněk ve stejném stavu.

Pro názornost si použití naší metafory ukážeme na jednoduchém příkladu. Mějme tedy nějaký systém. Abychom na něj mohli pohlížet jako na černou skříňku, musíme jej namodelovat v takové formě, která naplňuje rysy černé skříňky popsané výše (mj. si povšimněme, že tyto rysy nápadně odpovídají rozhraní mezi vnějším a vnitřním pohledem diskutovaném v oddílech 2.3 a 2.4, zejména potom čtveřici $\Sigma$, $\Phi$, $\Pi$, $\varphi$). Vhodnou formou jsou například právě přechodové systémy. Nechť je tedy jeho modelem následující přechodový systém




který popsané rysy černé skříňky zjevně naplňuje (resp. lze z něj vyčíst symboly tlačítek, zda reagují na stisk atd.). Podle uvedeného modelu již můžeme sestavit předpovědi

$$\{\beta\}\ \boldsymbol{\alpha}\ \{\alpha\}$$

$$\{\beta\}\ \boldsymbol{\alpha}\ \{\alpha\}\ \boldsymbol{\beta}\ \{\alpha, \beta\}$$

$$\{\beta\}\ \boldsymbol{\alpha}\ \{\alpha\}\ \boldsymbol{\beta}\ \{\alpha\}$$

$$\{\beta\}\ \boldsymbol{\alpha}\ \{\alpha\}\ \boldsymbol{\beta}\ \{\alpha\}\ \boldsymbol{\beta}\ \{\alpha, \beta\}$$

$$\{\beta\}\ \boldsymbol{\alpha}\ \{\alpha\}\ \boldsymbol{\beta}\ \{\alpha\}\ \boldsymbol{\beta}\ \{\alpha\}$$

$$\ldots$$

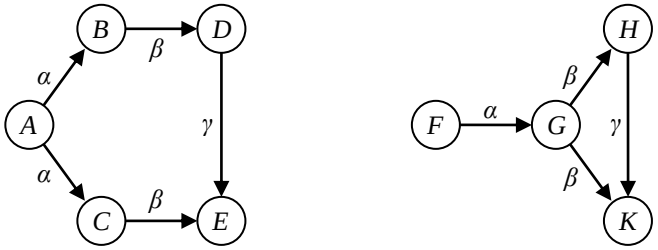
všech experimentů, které by pomyslný experimentátor zaznamenal. A právě tímto přeformulováním modelu na předpovědi experimentů jsme na daný systém pohlédli jako na výše popsanou černou skříňku. Pro úplnost dodejme, že ačkoliv je tvar našich předpovědí shodný se záznamem experimentů v metafoře prosvětlených tlačítek [2] a stejným způsobem je podle modelu také sestavujeme, tak jejich význam je z definice odlišný. Důsledkem je například nadbytečnost některých předpovědí (třetí, pátá atd.) v tomto příkladu, neboť jsou podmnožinami jiných předpovědí (druhé, čtvrté atd.).

V předchozích odstavcích jsme pomocí metafory především definovali co míníme ekvivalencí. Přesto však uvedenými zápisy a postupy nelze ekvivalenci formálně zcela uchopit (vždyť předpovědí experimentů může obecně být nekonečně mnoho a nemáme jiný způsob jak bychom je mohli zachytit konečným zápisem). Ostatně to ani nebylo účelem metafory, která je spíše myšlenkovou konstrukcí poskytující jednak jakýsi etalon ekvivalence a jednak určitější význam samotným přechodovým systémům (vždyť právě prostřednictvím významu se přechodový systém stává modelem).

## 5.3 Uchopení ekvivalence a porovnání

Po ujasnění co míníme ekvivalencí systémů, ztvárněném metaforou tlačítek s předpověďmi experimentů, se vrátíme k možným přístupům jak ekvivalenci formálně uchopit a porovnáme soulad těchto přístupů se zmíněnou metaforou. Konkrétně se budeme zabývat dvěma přístupy. Prvním přístupem je již zmíněná bisimulace, kterou Milner dodatečně zabudoval do teoretických základů svého kalkulu [8], a druhým přístupem je totožnost schopností, nad nimiž jsme vybudovali naší algebraickou strukturu (viz kapitola 4).

Abychom měli na čem vyzkoušet jak obstojí oba přístupy k uchopení ekvivalence, připravíme si nejdříve zajímavý příklad. Mějme tedy dva systémy, resp. jejich modely představované následujícími přechodovými systémy




(12)

na nichž si lze povšimnout, že jediným podstatným rozdílem je posun nedeterminismu (u levého přechodového systému nastane dříve, u pravého pak později). Pomocí metafory tedy ihned ověříme, zda jsou ekvivalentní či nikoliv. A zjistíme, že pro oba přechodové systémy jsou platné následující předpovědi experimentů

$$\{\beta, \gamma\} \ \boldsymbol{\alpha} \ \{\alpha, \gamma\}$$

$$\{\beta, \gamma\} \ \boldsymbol{\alpha} \ \{\alpha, \gamma\} \ \boldsymbol{\beta} \ \{\alpha, \beta\} \ \boldsymbol{\gamma} \ \{\alpha, \beta, \gamma\}$$

$$\{\beta, \gamma\} \ \boldsymbol{\alpha} \ \{\alpha, \gamma\} \ \boldsymbol{\beta} \ \{\alpha, \beta, \gamma\}$$

s tím, že nadbytečné předpovědi již zde byly vynechány. Jinými slovy, neexistuje tedy takový experiment, který by dokázal oba systémy rozlišit. A protože jsme řekli, že metafora bude pro soud o ekvivalenci určující, tak oba systémy jsou tímto prohlášeny za ekvivalentní.

**Bisimulace**

Koncept bisimulace zformuloval Park [9] krátce po prvním vydání Milnerova kalkulu, jak se z předmluvy dalšího vydání [8] můžeme dočíst. Milner tam doslova uvádí, že „nejdůležitějším pokrokem (v algebraickém přístupu ke komunikujícím systémům) je Parkův objev ideje bisimulace“ [8]. Následně Milner [8, 1] s využitím bisimulace přeformuloval teoretické základy související s rovnostmi výrazů v jeho kalkulu. Jak je samotná bisimulace a bisimulační ekvivalence definována si ukážeme v následujícím odstavci.

Ekvivalence založená na bisimulaci zohledňuje pouze strukturu větvení přechodových systémů právě tak, že se mohou vzájemně simulovat přechod po přechodu. Přesně řečeno, pro dané přechodové systémy $(S_1, \Lambda, R_1)$ a $(S_2, \Lambda, R_2)$ je bisimulace relací $M \subseteq S_1 \times S_2$ s následujícími vlastnostmi

$$\forall\,(X, Y) \in M\ :$$

$$\text{(a)}\quad \forall\,(X, \alpha, X') \in R_1 \quad \exists\,(Y, \alpha, Y') \in R_2 \quad : \quad (X', Y') \in M$$

$$\text{(b)}\quad \forall\,(Y, \alpha, Y') \in R_2 \quad \exists\,(X, \alpha, X') \in R_1 \quad : \quad (X', Y') \in M \qquad (13)$$

a říkáme, že přechodové systémy ve stavech $A_1 \in S_1$ a $A_2 \in S_2$ jsou bisimulačně ekvivalentní, právě když existuje bisimulace $M$ taková, že $(A_1, A_2) \in M$.

Milner [1] takto definovanou relaci $M$ označuje jako tzv. silnou bisimulaci a jak jsme výše předeslali, založil na ni také další druh rovnosti (zapisovanou symbolem ~). Modifikacemi silné bisimulace pak ještě vyvinul v zásadě dvě varianty, tzv. slabou bisimulaci (s rovností zapisovanou symbolem ≈) a tzv. ekvivalenci (s rovností zapisovanou symbolem =). Posledně jmenovaná tedy nahrazuje zcela nevyhovující isomorfnost derivačních stromů a Milner pomocí ní také ověřil platnost vlastností operací kalkulu. Protože se všechny tři varianty liší pouze v různém zacházení s tzv. tichou akcí, která se však v námi zvoleném výřezu základního kalkulu nevyskytuje, tak nám zde splývají do bisimulace jediné.

Nyní již ověříme, zda přechodové systémy z výše připraveného příkladu (12) jsou bisimulačně ekvivalentní. Podle výše uvedené definice (13) je nezbytné najít bisimulaci $M$, takže pro urychlení rovnou vyzkoušíme následující relaci

$$\{\ (A, F),\ \mathbf{(B, G)},\ \mathbf{(C, G)},\ (D, H),\ (E, K)\ \}$$

a zjišťujeme, že dvojice $(B, G)$ a $(C, G)$ nesplňují podmínku (b) z definice, protože v relaci chybí dvojice $(D, K)$ a $(E, H)$. Vyzkoušíme tedy relaci rozšířenou o chybějící dvojice

$$\{\ (A, F),\ (B, G),\ (C, G),\ (D, H),\ (E, K),\ \mathbf{(D, K)},\ \mathbf{(E, H)}\ \}$$

a tentokrát zjišťujeme, že nově přidané dvojice $(D, K)$ a $(E, H)$ nesplňují podmínky (a) a (b), v uvedeném pořadí. Tím se dostáváme do sporu, což znamená, že bisimulace obsahující dvojici $(A, F)$ nemůže být nalezena. Systémy tedy bisimulačně ekvivalentní nejsou.

Ovšem to potom znamená, že bisimulace obecně neodpovídá naší metafoře. Přitom Milner výslovně uvádí, že „jakékoliv dva systémy (v daných stavech) chceme rozlišovat pouze když odlišnost může být zjištěna nějakým dalším systémem při interakci s každým z nich“ [1] a sám použil myšlenkový experiment odpovídající naší metafoře tlačítek, kterým pak potřebu bisimulace odůvodnil. Takže bisimulace neodpovídá nejen naší metafoře, ale ani původnímu záměru Milnera. Posunem od derivačních stromů k bisimulaci se Milner sice svému záměru ohledně rozlišitelnosti (a tedy ekvivalence) systémů velmi přiblížil, ale bisimulace je zjevně stále příliš jemným pojetím ekvivalence (resp. rozlišuje systémy stále příliš jemně).

Je pravda, že zde posuzované přechodové systémy (12) by nebyly ekvivalentní v rámci již také zmíněné metafory tlačítek s duplikátorem. Bisimulace by tedy v daném případě takové metafoře odpovídala. Avšak Bloom a Meyer [2] uvádí příklad dvou jiných přechodových systémů, kde se metafora tlačítek s duplikátorem a bisimulace již rozchází. Takže v důsledku vyvstává dokonce otázka, jaké metafoře bisimulace vlastně odpovídá (resp. v jakém smyslu jsou ekvivalentní dva systémy, pro které existuje bisimulace). Ani Bloom a Meyer [2] však na tuto otázku jednoznačně neodpověděli.

Nakonec si ještě ukážeme, jak by se posuzované přechodové systémy (12) vyjádřili v Milnerově základním kalkulu. Stačí si uvědomit, že každý přechodový systém buď je fragmentem sémantického přechodového systému odpovídajícím nějakému výrazu z množiny $\mathcal{Z}$, anebo může být z takových fragmentů díky definičním rovnostem pospojován (podrobněji viz oddíl 3.3). Potom snadno dospějeme ke dvěma výrazům

$$\alpha\cdot\beta\cdot\gamma\cdot\mathbf{0} + \alpha\cdot\beta\cdot\mathbf{0} \qquad \neq \qquad \alpha\cdot(\beta\cdot\gamma\cdot\mathbf{0} + \beta\cdot\mathbf{0}) \tag{14}$$

o kterých zároveň víme, že si nejsou rovny. Ovšem, vždyť konečným pojetím ekvivalence výrazů v rámci Milnerova základního kalkulu je bisimulační ekvivalence jim odpovídajících přechodových systémů; a my jsme výše již ověřili, že přechodové systémy (12) bisimulačně ekvivalentní nejsou. Povšimněme si ještě, že rovnost uvedených výrazů by z algebraického hlediska jistě zaručila distributivita. Potom je zákonité, že Milner [1] výslovně upozorňuje na neplatnost distributivity v jeho kalkulu.

Po sporných derivačních stromech tedy ani bisimulace nedovolí výrazům v Milnerově základním kalkulu, aby měly význam vnějšího projevu odpovídajících přechodových systémů (resp. projevu formalizovaného v oddílu 5.2 pomocí předpovědí experimentů). Jestliže totiž přechodovým systémům (12) se stejným vnějším projevem odpovídají vzájemně neekvivalentní výrazy, pak tyto výrazy význam vnějšího projevu mít nemohou.

**Totožnost schopností**

Pojem schopnost jsme zavedli pro formální popis vnějšího projevu (viz oddíl 4.1), který je algebraicky snadno uchopitelný a který jsme od počátku konstruovali tak, aby byl oproštěn od nadbytečné informace nesené (vnitřní) strukturou přechodového systému.

Ekvivalence založená na schopnostech je pak indukována jednoduše jejich totožností. Přesně řečeno, dané přechodové systémy $(S_1, \Lambda, R_1)$ a $(S_2, \Lambda, R_2)$ ve stavech $A_1 \in S_1$ a $A_2 \in S_2$ jsou ekvivalentní, právě když jejich schopnosti $\Pi^{\Psi}{}_{A_1}$ a $\Pi^{\Psi}{}_{A_2}$ jsou totožné.

Nyní již ověříme, zda přechodové systémy z výše připraveného příkladu (12) jsou ekvivalentní ve smyslu schopností. Nejdříve musíme odvodit množiny $\mathcal{E}^{\Psi}{}_{A}$ a $\mathcal{E}^{\Psi}{}_{F}$ všech přípustných provedení ze stavů $A$ a $F$. Výslednými množinami jsou

$$\mathcal{E}^{\Psi}{}_{A} = \{ \ (A, \alpha^{\beta}, B)\ (B, \beta^{\gamma}, D)\ (D, \gamma^{\varnothing}, E)\ ,\ \ (A, \alpha^{\beta}, C)\ (C, \beta^{\varnothing}, E)\ \}$$

$$\mathcal{E}^{\Psi}{}_{F} = \{ \ (F, \alpha^{\beta}, G)\ (G, \beta^{\gamma}, H)\ (H, \gamma^{\varnothing}, K)\ ,\ \ (F, \alpha^{\beta}, G)\ (G, \beta^{\varnothing}, K)\ \}$$

ze kterých pak už snadno odvodíme příslušné schopnosti $\Pi^{\Psi}{}_{A}$ a $\Pi^{\Psi}{}_{F}$ (tj. množiny přípustných posloupností interpretací)

$$\Pi^{\Psi}{}_{A} = \{\ \alpha^{\beta}\beta^{\gamma}\gamma^{\varnothing},\ \alpha^{\beta}\beta^{\varnothing}\ \} \qquad\qquad \Pi^{\Psi}{}_{F} = \{\ \alpha^{\beta}\beta^{\gamma}\gamma^{\varnothing},\ \alpha^{\beta}\beta^{\varnothing}\ \}$$

a zjišťujeme, že jsou totožné. Takže ve smyslu schopností systémy ekvivalentní jsou. Oproti bisimulaci tedy totožnost schopností v tomto případě souhlasí s naší metaforou tlačítek (ale obecný soulad to samozřejmě nedokazuje, to bude předmětem další práce).

Nakonec si ukážeme, jak by se posuzované přechodové systémy (12) vyjádřili v rámci navržené algebraické struktury (viz oddíl 4.2). Stačí si uvědomit, že každému výrazu (tzn. také

podvýrazu) vyjadřujícímu schopnost vždy odpovídá určitý stav příslušného přechodového systému, přičemž operandům součtu (+) odpovídá stav stejný jako součtu samému (podrobněji viz oddíl 4.2). Potom snadno dospějeme ke dvěma výrazům

$$\alpha{\cdot}\beta{\cdot}\gamma{\cdot}\mathbf{0} + \alpha{\cdot}\beta{\cdot}\mathbf{0} \qquad = \qquad \alpha{\cdot}(\beta{\cdot}\gamma{\cdot}\mathbf{0} + \beta{\cdot}\mathbf{0}) \tag{15}$$

které by si měly být rovny. Ovšem, neboť v rámci navržené algebraické struktury je rovnost chápána jako totožnost prvků nosné množiny *K*, resp. totožnost schopností, vyjadřovaných výrazy na obou jejích stranách; a uvedené výrazy by měly vyjadřovat právě schopnosti $\Pi^{\Psi}{}_{A}$ a $\Pi^{\Psi}{}_{F}$, které jsme výše odvodili z výrazům odpovídajících přechodových systémů (12). Jinými slovy, uvedená rovnost by vlastně měla vyjadřovat

$$\Pi^{\Psi}{}_{A} \qquad = \qquad \Pi^{\Psi}{}_{F}$$

což ověříme vyhodnocením výrazů

$$\begin{array}{ccc}
\alpha \cdot \beta \cdot \gamma \cdot \mathbf{0} \;+\; \alpha \cdot \beta \cdot \mathbf{0} & = & \alpha \cdot (\beta \cdot \gamma \cdot \mathbf{0} \;+\; \beta \cdot \mathbf{0}) \\
\alpha \cdot \beta \cdot \gamma \cdot \varnothing \;+\; \alpha \cdot \beta \cdot \varnothing & = & \alpha \cdot (\beta \cdot \gamma \cdot \varnothing \;+\; \beta \cdot \varnothing) \\
\alpha \cdot \beta \cdot \{\, \gamma^{\varnothing} \,\} \;+\; \alpha \cdot \{\, \beta^{\varnothing} \,\} & = & \alpha \cdot (\beta \cdot \{\, \gamma^{\varnothing} \,\} \;+\; \{\, \beta^{\varnothing} \,\}) \\
\alpha \cdot \{\, \beta^{\gamma}\gamma^{\varnothing} \,\} \;+\; \{\, \alpha^{\beta}\beta^{\varnothing} \,\} & = & \alpha \cdot (\{\, \beta^{\gamma}\gamma^{\varnothing} \,\} \;+\; \{\, \beta^{\varnothing} \,\}) \\
\{\, \alpha^{\beta}\beta^{\gamma}\gamma^{\varnothing} \,\} \;+\; \{\, \alpha^{\beta}\beta^{\varnothing} \,\} & = & \alpha \cdot \{\, \beta^{\gamma}\gamma^{\varnothing}, \beta^{\varnothing} \,\} \\
\{\, \alpha^{\beta}\beta^{\gamma}\gamma^{\varnothing}, \alpha^{\beta}\beta^{\varnothing} \,\} & = & \{\, \alpha^{\beta}\beta^{\gamma}\gamma^{\varnothing}, \alpha^{\beta}\beta^{\varnothing} \,\}
\end{array}$$

a zjišťujeme, že tomu tak skutečně je. Uvedené výrazy (15) tedy podle očekávání vyjadřují výše odvozené (totožné) schopnosti $\Pi^{\Psi}{}_{A}$ a $\Pi^{\Psi}{}_{F}$ přechodových systémů (12). Takže uvedené výrazy (15) skutečně jsou ekvivalentní (tzn. jsou si rovny). Navíc se ukazuje, že v navržené algebraické struktuře (oproti Milnerově kalkulu) nějaká forma distributivity platí. Nebude totiž těžké ukázat, že místo podvýrazů $\gamma{\cdot}\mathbf{0}$ a $\mathbf{0}$ v rovnosti (15) mohou vystupovat libovolné schopnosti.

Příklad s přechodovými systémy (12) názorně potvrzuje rozhodující vliv ekvivalence na význam výrazů. Pro odpovídající přechodový systém mají výrazy (14) a (15) naprosto stejný tvar a přesto mají odlišný význam. Ekvivalence tedy rozhoduje nejen o vzájemné nahraditelnosti systémů či výrazů, ale rozhoduje dokonce o tom, nad čím přesně pomocí budovaného formálního aparátu (ať už algebraického nebo jiného) vlastně budeme uvažovat. Jinými slovy, pokud dané formální pojetí ekvivalence nebude věrně odrážet skutečnost, tak by se nám mohlo stát, že budeme formálně uvažovat o něčem (více či méně) jiném, než bychom se domnívali (byť by dané pojetí ekvivalence bylo „jen“ jemnější než v modelované skutečnosti).

Nyní již víme, že z hlediska vnějšího projevu je bisimulace příliš jemná (resp. rozlišuje i „něco“, co pouhou interakcí systémů prostřednictvím akcí ve skutečnosti rozlišit nelze). Ostatně bisimulační ekvivalence „je uznávána jakožto nejjemnější ekvivalence chování, o které lze u souběžných systémů uvažovat“ [10]. Naopak u ekvivalence založené na totožnosti schopností se lze (na základě dosavadního výzkumu) domnívat, že její jemnost je z hlediska vnějšího projevu právě na „správné“ hranici. Důležité tedy bude v další práci obecně ukázat (pomocí metafory tlačítek či jinak), že neexistuje jemnější ekvivalence, ve které by každé dva nerozlišitelné systémy (tj. systémy s nerozlišitelným vnějším projevem) byly ještě ekvivalentní.

# 6 Závěr

**Stručná rekapitulace**

Lze soudit, že paradigma multiagentních systémů by uplatnění v informačních systémech (a obecně v softwarovém inženýrství) najít mohlo. Přínosem by potom mohlo být především v případě komplexních nebo přirozeně distribuovaných systémů. Avšak při návrhu multiagentních systémů nelze využít žádné ze stávajících metod softwarového inženýrství, které jsou v důsledku všechny založeny na algoritmické dekompozici. Vždyť komponenty těchto systémů (agenty) jsou navrhovány s lokálním úhlem pohledu, což vede k jejich volnějším vzájemným vazbám a větší autonomii. Neexistuje tedy globální (byť distribuovaný či paralelizovaný) algoritmus. Funkce multiagentního systému je dána dynamikou (tzn. možnými souběhy) interakcí mezi agenty. Navíc ověřování správnosti (a vlastně jakékoliv vyšetřování) této dynamiky pomocí testování (tj. obvyklou technikou v softwarovém inženýrství) je zjevně vyloučené. Dokud tedy nebudeme mít adekvátní metodu pro návrh i vyšetřování, tak veškeré snahy o multiagentní systémy (pokud vynecháme uplatnění v simulacích) budou mít podobu spíše klasických distribuovaných (či paralelních) systémů, což je důsledkem pochopitelné opatrnosti.

Přirozeně se zde nabízí hledat pomoc v odvětví formálních metod. Ačkoliv úkoly, před které je toto odvětví stavěno, se ukazují jako nečekaně obtížné, tak již bylo dosaženo řady slibných výsledků. Svoji pozornost jsme soustředili na Milnerův kalkulus (CCS), protože byl vyvíjen s velmi podobnou motivací (resp. snahou obecněji a plně postihnout pozorovatelné chování souběžných systémů) a jeví se vhodným východiskem pro nalezení aparátu, který by mohl být základem adekvátní metody pro modelování (a tím návrh i vyšetřování) multiagentních systémů. Avšak ekvivalence v Milnerovo kalkulu, tj. bisimulační ekvivalence přechodových systémů, je příliš jemná. Jak jsme si ukázali (viz oddíl 5.3), bisimulační ekvivalence rozliší i takové přechodové systémy, které mají totožný vnější projev. Pro připomenutí dodejme, že pojem vnější projev jsme intuitivně definovali jako pozorovatelné chování, kdy vnější pozorovatel smí s přechodovým systémem manipulovat jen interakcemi prostřednictvím akcí (viz kapitola 0), a později jsme pojem upřesnili pomocí metafory tlačítek (viz oddíl 5.2). Přitom již v úvodu jsme si ujasnili, že pro formální uvažování (což je smyslem kalkulu) je problémem i pokud ekvivalence neodpovídá zamýšlenému pojetí chování „pouze“ kvůli své větší jemnosti (viz kapitola 0). Stručně řečeno, Milnerovo kalkulem (s bisimulační ekvivalencí) bychom formálně neuvažovali nad vnějším projevem přechodových systémů, ale nad nějakým blíže neurčeným pozorovatelným chováním. A blíže neurčeným proto, že pozorovatelné chování (implicitně definované bisimulační ekvivalencí) je obtížné explicitně charakterizovat; vždyť dosud se nepodařilo nalézt ani metaforu, která by bisimulační ekvivalenci odpovídala (jak je zmíněno v oddílu 5.3). Takže právě přílišná jemnost bisimulační ekvivalence byla důvodem záměru vpravit do Milnerovo kalkulu, resp. do jeho operačně sémantického modelu, jinou (patřičně hrubší) ekvivalenci.

Milnerovo kalkulus opravdu lze chápat jako rámec pro rodinu (operačně sémantických) modelů definovaných takovými ekvivalencemi, jejichž podmnožinou je bisimulační ekvivalence

(viz citace Hoara v kapitole 0). Avšak Milnerův způsob budování kalkulu dovoluje vpravit jen takové ekvivalence, které jsou indukovány existencí nějakého vztahu „podobnosti stavů“ (např. existencí bisimulace). Milner při budování kalkulu totiž bezprostředně navazuje topologii přechodového grafu na syntaxi výrazů tak, že každému výrazu odpovídá právě jeden stav (viz oddíl 3.3), což je pak souhrnně formalizováno přechodovými pravidly (např. výraz $\alpha\cdot\beta\cdot\gamma\cdot\mathbf{0} + \alpha\cdot\beta\cdot\mathbf{0}$ se tak provedením akce $\alpha$ nemůže proměnit na výraz $\beta\cdot\gamma\cdot\mathbf{0} + \beta\cdot\mathbf{0}$, neboť takovému výrazu neodpovídá žádný stav přechodového grafu). A přechodovým pravidlům se ekvivalence musí podřídit (např. výraz $\alpha\cdot\beta\cdot\gamma\cdot\mathbf{0} + \alpha\cdot\beta\cdot\mathbf{0}$ tedy nesmí být ekvivalentní s výrazem $\alpha\cdot(\beta\cdot\gamma\cdot\mathbf{0} + \beta\cdot\mathbf{0})$, neboť podle přechodových pravidel se provedením stejné akce $\alpha$ oba výrazy promění na takové výrazy, které již ekvivalentní jistě nejsou – alternativně řečeno, odpovídají „nepodobným“ stavům). Takže Milnerův způsob budování kalkulu a především zmíněná přechodová pravidla příliš omezují volbu ekvivalence (např. právě ekvivalentnost uvedených výrazů $\alpha\cdot\beta\cdot\gamma\cdot\mathbf{0} + \alpha\cdot\beta\cdot\mathbf{0}$ a $\alpha\cdot(\beta\cdot\gamma\cdot\mathbf{0} + \beta\cdot\mathbf{0})$ by totiž byla žádoucí, jak jsme zjistili v oddílu 5.3). Milnerovo způsobem tedy kalkulus s patřičně hrubší ekvivalencí vybudovat nelze.

Tento problém i jeho příčinu jsme přitom předpověděli již při studiu vnitřní struktury přechodových systémů (viz oddíl 2.4). Tam jsme dospěli ke zjištění, že (pro vyjádření vnějšího projevu) nadbytečná informace nesená vnitřní strukturou nespočívá jen v pojmenování stavů, ale zejména v topologii přechodového grafu (a to i pokud neobsahuje žádné ekvivalentní stavy). Jak jsme však zmínili v předchozím odstavci, tak v Milnerovo přístupu je kompletní informace o topologii nejen přenesena na syntaxi výrazů kalkulu, ale (což je podstatnější) je plně respektována ve formě přechodových pravidel (jež ekvivalence nemůže překročit). Proto jsme se zcela vyhnuli přímému použití přechodových grafů a zvolili následující přístup.

Z přechodových systémů, resp. jejich přípustných provedení (tzn. posloupností přechodů), jsme odvodili formální popis vnějšího projevu (viz oddíl 4.1), který jsme označili pojmem schopnost (neboť vystihuje, jak je přechodový systém schopen interagovat se svým okolím prostřednictvím akcí). Přičemž klíčovým bylo uvědomění si a definování pojmu interpretace akce, který zahrnuje kompletní informaci o důsledku dotyčné akce na následující schopnost přechodového systému interagovat. Vždyť explicitní definování vnějšího projevu formou schopnosti je umožněno právě objevem, že ke každému přechodovému systému existuje takový přechodový systém se stejnou schopností, který však ve smyslu interpretací akcí je deterministický. Také jsme si ukázali (viz oddíl 4.1), že ty schopnosti, kterých může nabývat přechodový systém, vlastně reprezentují jeho stavy rozlišitelné vnějším pozorovatelem. Dále jsme schopnostmi indukovanou ekvivalenci porovnali na jednom (ovšem charakteristickém) příkladu jak s bisimulační ekvivalencí, tak s metaforou tlačítek (viz oddíl 5.3). Potvrdili jsme si tím, že schopnostmi indukovaná ekvivalence přechodových systémů je dostatečně hrubá, aby jí odpovídající pozorovatelné chování (zde schopnost) bylo vnějším projevem. Avšak ta skutečnost, že neexistuje žádná jemnější ekvivalence, která by přes svou vyšší jemnost stále zohledňovala jen odlišnosti ve vnějším projevu, tím potvrzena samozřejmě nebyla (potřebný důkaz bude předmětem další práce). Nakonec, rozuměno po zavedení pojmu schopnost (jež zároveň reprezentuje i třídu ekvivalence přechodových systémů), jsme teprve vybudovali složenou algebraickou strukturu (viz oddíl 4.2). Nevybudovali jsme ji tedy nad množinou výrazů nebo přechodových systémů (jak je postupováno v případě Milnerovo kalkulu), ale právě nad množinou schopností. Přičemž algebraické operace byly definovány tak, aby jejich signatura zajistila stejný jazyk pro zápis výrazů jaký má Milnerův kalkulus. Také vlastnosti (či axiomy) navržené algebraické struktury zahrnují veškeré vlastnosti Milnerovo kalkulu a

dokonce je doplňují o další vlastnost (viz určitá forma distributivity zmíněná koncem oddílu 5.3). Stejný jazyk výrazů a pouhé rozšíření sady vlastností zde jasně poukazují na Hoarovo slova o tom, že Milnerovo kalkulus je rámcem pro operačně sémantické modely definované bisimulační či hrubší ekvivalencí. Navržená algebraická struktura tedy Milnerovo kalkulus upřesňuje.

Zvolený přístup přináší kromě patřičného upřesnění Milnerovo kalkulu i některé další výhody. Například je automaticky zajištěno, že ekvivalence přechodových systémů je zároveň kongruencí (neboť algebraické operace jsou definovány teprve nad schopnostmi, tedy nad třídami ekvivalence). Dále není třeba zavádět speciální definiční rovnost (neboť si vystačíme se standardní rovností – např. význam symbolu $\boldsymbol{A}$ lze definovat rovnicí $\boldsymbol{A} = \alpha \cdot \boldsymbol{A}$). Nicméně především, význam výrazů lze snáze vymezit a formálně o něm uvažovat (neboť existuje jeho explicitní, formálně uchopitelný popis – zde konkrétně v podobě schopnosti). Naproti tomu význam výrazů v Milnerovo kalkulu, tj. pozorovatelné chování (definované implicitně bisimulační ekvivalencí), je prakticky nezřetelný. Jinými slovy, je obtížné říci, co přesně je (a co už není) na přechodových systémech podle Milnera „pozorovatelné“ (zjevně to však neodpovídá Milnerovo výchozí tezi, že pozorovat systém znamená komunikovat s ním).

**Směr další práce**

Je zřejmé, že pro nalezení či vybudování aparátu, jenž by se stal základem adekvátní metody pro modelování multiagentních systémů, je klíčovým problémem ekvivalence. Vždyť právě ekvivalence rozhoduje o tom, co je „chování“ (tj. nad čím tedy budeme pomocí výsledného aparátu formálně uvažovat) a jak přiléhavé mohou modely být. V disertační práci pak určitě bude vhodné navrženou ekvivalenci (indukovanou zavedenými schopnostmi) porovnat také s dalšími ekvivalencemi (jde zejména o *ready simulation* a *ready traces* ekvivalence, které by se svou jemností mohly pohybovat okolo navržené ekvivalence). Nicméně, stěžejní výsledek disertační práce nebude spočívat v navržené ekvivalenci (a dopracování jejího odvození) ani ve zmíněném porovnání.

Hlavním cílem disertační práce bude dokázat, že navržená ekvivalence je nejjemnější možnou ekvivalencí přechodových systémů z hlediska jejich vnějšího projevu. Jinými slovy, že neexistuje žádná jemnější ekvivalence, která by stále zohledňovala jen odlišnosti zjistitelné interakcemi (či manipulacemi) s přechodovým systémem pouze prostřednictvím atomických akcí (tzn. bez pomoci duplikací, ovlivňování nedeterminismu a bez jiných manipulací). Tím potvrdíme, že se nám podařilo najít takovou formalizaci ekvivalence, která přesně odpovídá tomu, jak chápeme „chování“ (zde vymezené pojmem vnější projev). Přitom to, jak „chování“ chápeme, je z praktického hlediska jednoduše charakterizovatelné (viz zmíněné omezení pouze na interakce prostřednictvím akcí) a díky tomu půjde výsledky práce snadno vztáhnout na poměrně širokou třídu aplikací (tedy nejen na multiagentní systémy).

Druhým (doprovodným) cílem disertační práce pak bude preciznější vybudování (zde navržené) algebraické struktury tak, aby byla lépe zřejmá zejména její konzistence s nezávisle formalizovanou (a v rámci hlavního cíle disertační práce ověřenou) ekvivalencí. Jak již bylo zmíněno, právě taková algebraická struktura patřičně upřesňuje Milnerův kalkulus a může se stát základem potřebné adekvátní metody pro modelování (nejen) multiagentních systémů.

# Literatura


[1] Milner, R., *Communication and Concurrency*, Prentice Hall, Harlow, England, 1989. ISBN 0-13-114984-9.

[2] Bloom, B., Meyer, A.R., "Experimenting with process equivalence", *Theoretical Computer Science*, Volume 101, Issue 2, Elsevier, Essex, 1992. (pp. 223-237)

[3] Broy, M., *Characterizing the Behavior of Reactive Systems by Trace Sets*, Technical Report TUM-I9102, Institut für Informatik, Technische Universität München, 1995.

[4] Hoare, C.A.R., *Communicating Sequential Processes*, Prentice Hall, Englewood Cliffs, New Jersey, 1985 (reissue 2004). ISBN 0-13-153289-8.

[5] Thomas, W., *Applied Automata Theory*, Lecture Notes, RWTH Aachen, 2005. URL: <http://drona.csa.iisc.ernet.in/~deepakd/atc-common/wolfgang-aat.pdf> [cit. 2010-8-15].

[6] Holcombe, M., *Algebraic Automata Theory*, Cambridge University Press, Cambridge, 2004. ISBN 0-521-60492-3.

[7] Hopcroft, J.E., Motwani, R., Ullman, J.D., *Introduction to Automata Theory, Languages, and Computation (2nd edition)*, Addison-Wesley, USA, 2001. ISBN 0-201-44124-1.

[8] Milner, R., *A Calculus of Communicating Systems*, LFCS Report ECS-LFCS-86-7, Department of Computer Science, University of Edinburgh, 1986.

[9] Park, D., "Concurrency and automata on infinite sequences", *Theoretical Computer Science*, Volume 104, Springer, Berlin, 1981. (pp. 167-183)

[10] Rabinovich, A., "Checking equivalences between concurrent systems of finite agents (Extended abstract)", *Automata, Languages and Programming*, Volume 623, Springer, Berlin, 1992. (pp. 696-707)

[11] Štěpánková, O., Mařík, V., Lhotská, L., „Distribuovaná umělá inteligence", *Umělá inteligence (2)*, Academia, Praha, 1997. ISBN 80-200-0504-8. (s. 142-177)

[12] Mařík, V., Pěchouček, M., Štěpánková, O., „Multiagentní systémy: Principy komunikace a základní formální architektury", *Umělá inteligence (3)*, Academia, Praha, 2001. ISBN 80-200-0472-6. (s. 189-236)

[13] Ferber, J., *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*, Addison-Wesley, Harlow, 1999. ISBN 0-201-36048-9.

[14] Smith, J.D., Phillips, D.M., *Interoperable Acquisition for Systems of Systems: The Challenges*, Technical Note CMU/SEI-2006-TN-034, Software Engineering Institute, Carnegie Mellon University, 2006.

[15] Martí, I., at al., "A multi-agent system for managing adverse weather situations on the road network", *Integrated Computer-Aided Engineering*, Volume 17, Issue 2, IOS Press, 2010. (pp. 145-155)

[16] Baier, C., Katoen, J.P., *Principles of Model Checking*, MIT Press, Cambridge, Massachusetts, 2008. ISBN 978-0-262-02649-9.

[17] Parnas, D.L., "Really Rethinking 'Formal Methods'", *Computer*, Volume 43, Issue 1, IEEE Computer Society Press, 2010. (pp. 28-34)

[18] Plaks, T., Pike, L., Parnas, D.L., "Letters to the Editor: Rethinking Formal Methods", *Computer*, Volume 43, Issue 3, IEEE Computer Society Press, 2010. (pp. 6-9)

[19] Johnson, S.C., Butler, R.W., "Formal Methods", *The Avionics Handbook*, CRC Press, Boca Raton, 2001. ISBN 0-8493-8348-X. (chapter 21)

[20] Abramson, D., Pike, L., "When Formal Systems Kill: Computer Ethics and Formal Methods", *APA Newsletter on Philosophy and Computers*, Volume 11, Issue 1, The American Philosophical Association, 2011. (pp. 6-13)

[21] Pinkas, P., Klečková, J., "Modeling of Capabilities of a System in Algebraic Manner", *1st International Conference on Networked Digital Technologies (NDT 2009)*, Ostrava, 2009. ISBN 978-1-4244-4614-8. (pp. 458-463)

[22] Pinkas, P., Klečková, J., "The Composite Algebraic Structure over Capabilities and Actions of a System", *International Conference for Internet Technology and Secured Transactions (ICITST 2009)*, London, 2009. ISBN 978-1-4244-5647-5. (pp. 177-182)