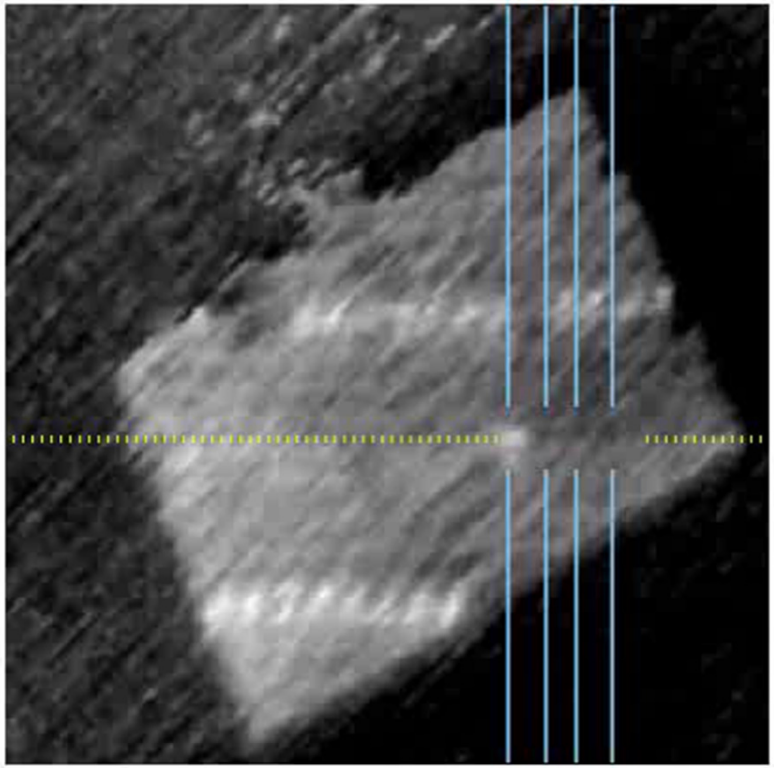
UNIVERSITY OF OXFORD

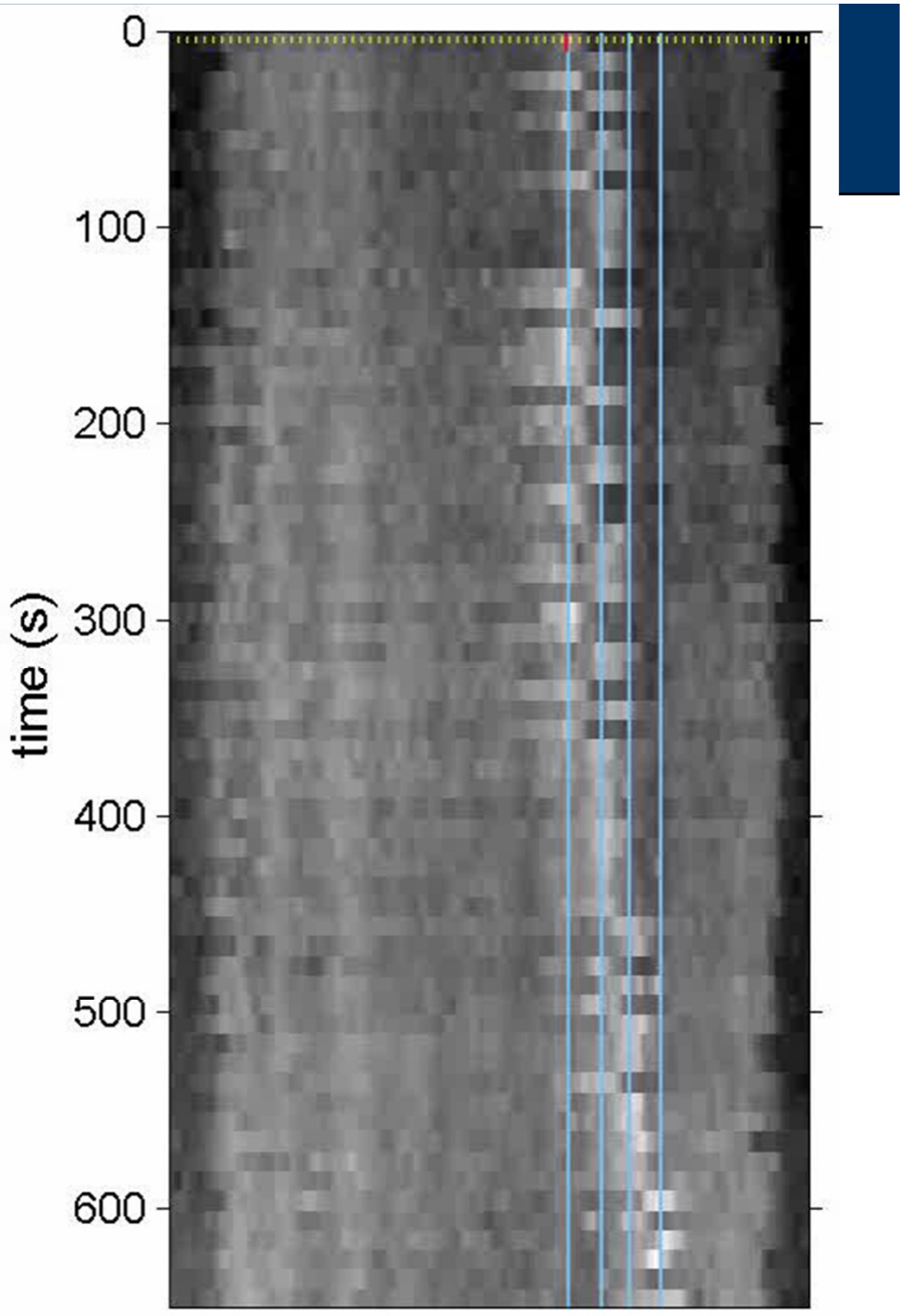
On Quantitative Modelling and Verification of DNA Walker Circuits Using Stochastic Petri Nets

Marta Kwiatkowska, University of Oxford

PN/ACSD 2015, Brussels

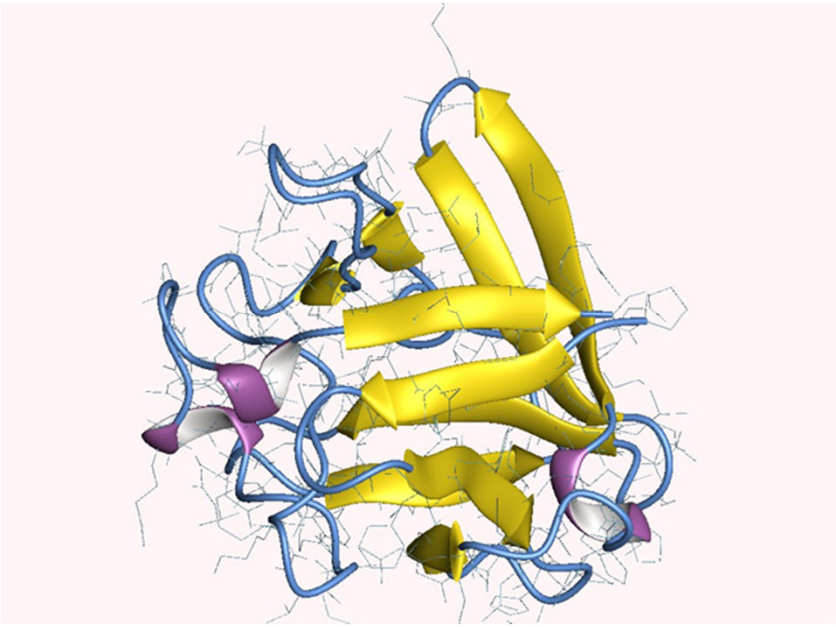
Image Frame t = 10s



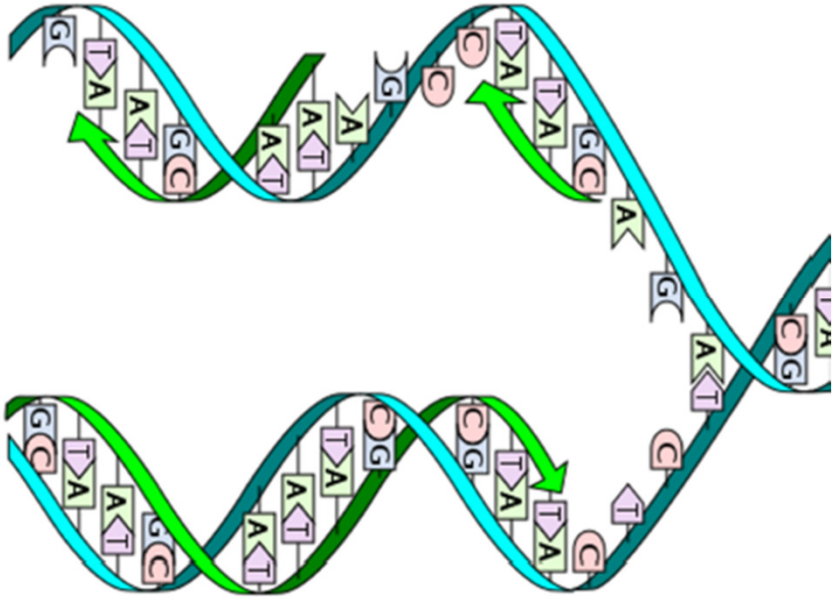


At the nanoscale…

- The world of molecules



Human FGF protein



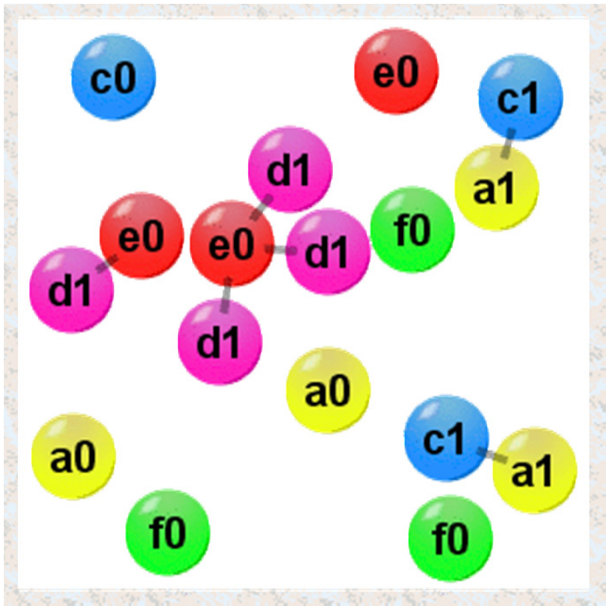
width 2nm

DNA: versatile, easy to synthesize

Molecular programming

- The application of computational concepts and design methods to nanotechnology, esp biochemical systems

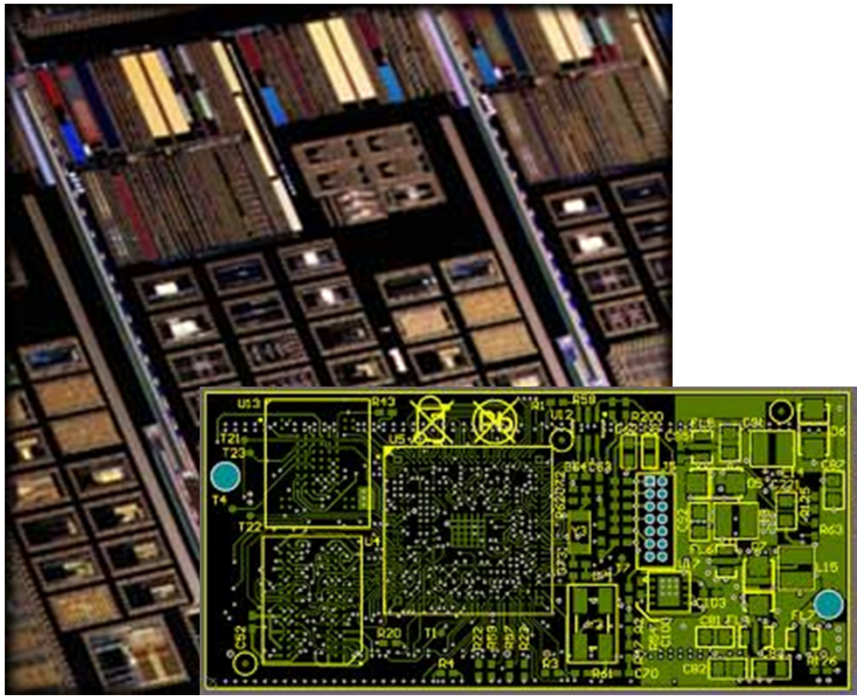
- Molecular programs are
  - networks of molecules
  - can interact
  - can move

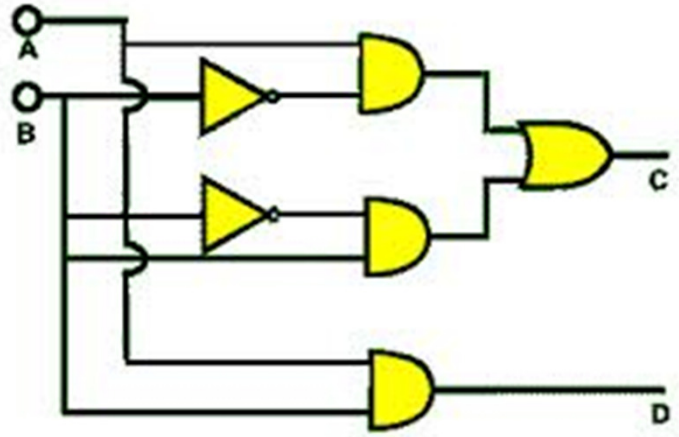


- Key observation
  - can store/process information
  - are programmable
  - (can compute a desired outcome)
  - proceed autonomously

- Petri nets are particularly appropriate!

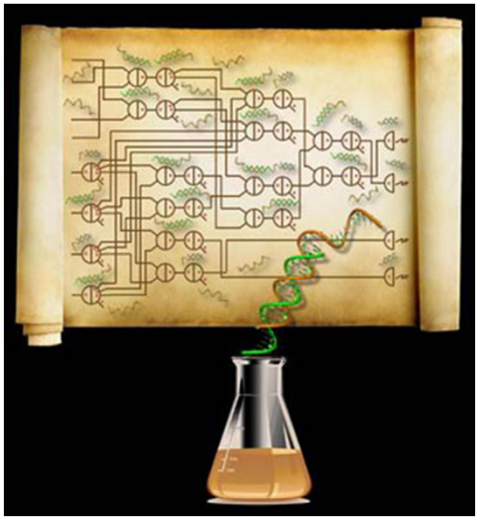
Digital circuits





- Logic gates realised in silicon
- 0s and 1s are represented as low and high voltage
- Hardware verification indispensable as design methodology

DNA circuits, in solution

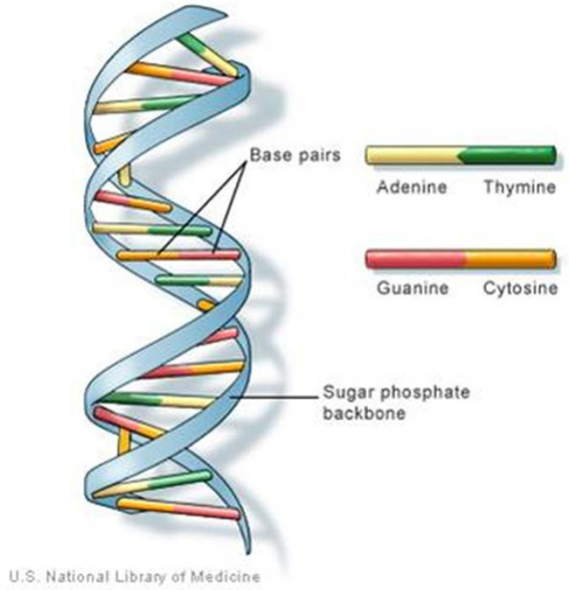


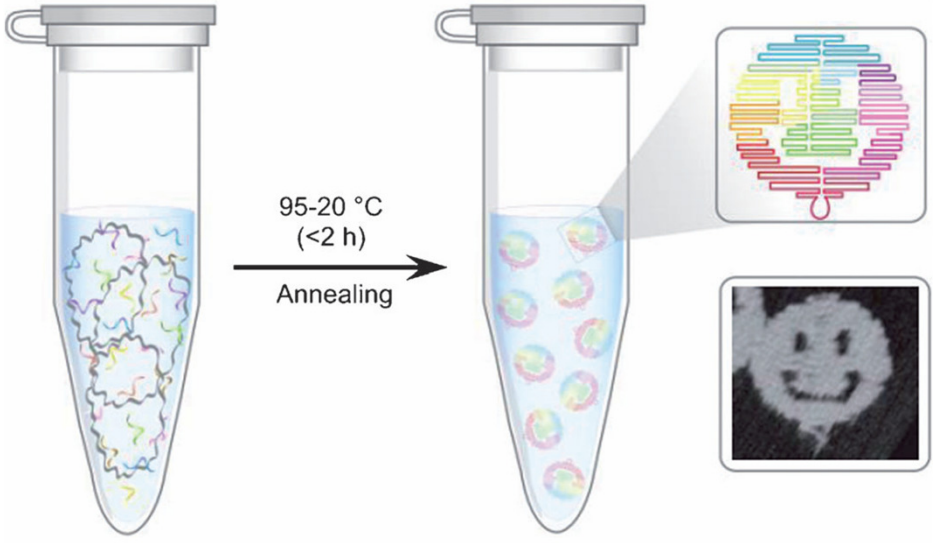
[Qian, Winfree, *Science* 2012]

- "Computing with soup" (The Economist 2012)
- Single strands are inputs and outputs
- Circuit of 130 strands computes square root of 4 bit number, rounded down
- 10 hours, but it's a first…

Pop quiz, hotshot: what's the square root of 13?
Science Photo Library/Alamy

DNA nanostructures



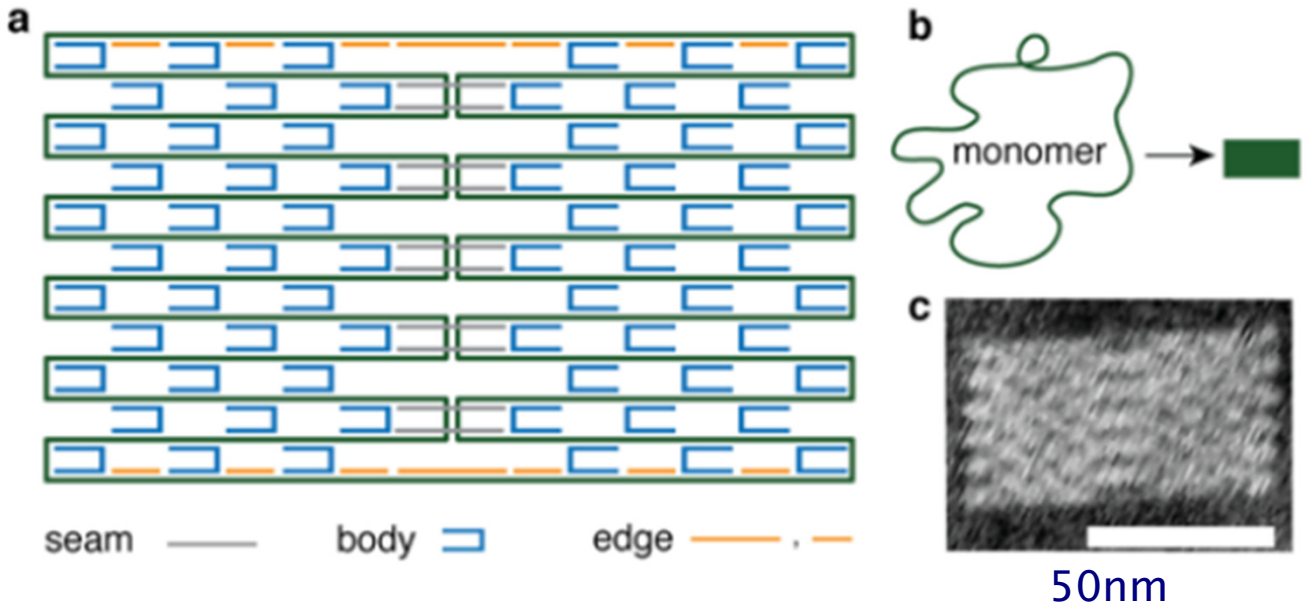


DNA origami

- DNA origami [Rothemund, *Nature* 2006]
  - DNA can self-assemble into structures – "molecular IKEA?"
  - programmable self-assembly (can form tiles, nanotubes, boxes that can open, etc)
  - simple manufacturing process (heating and cooling), not yet well understood

DNA origami tiles

- Origami tiles made from DNA [Turberfield lab]



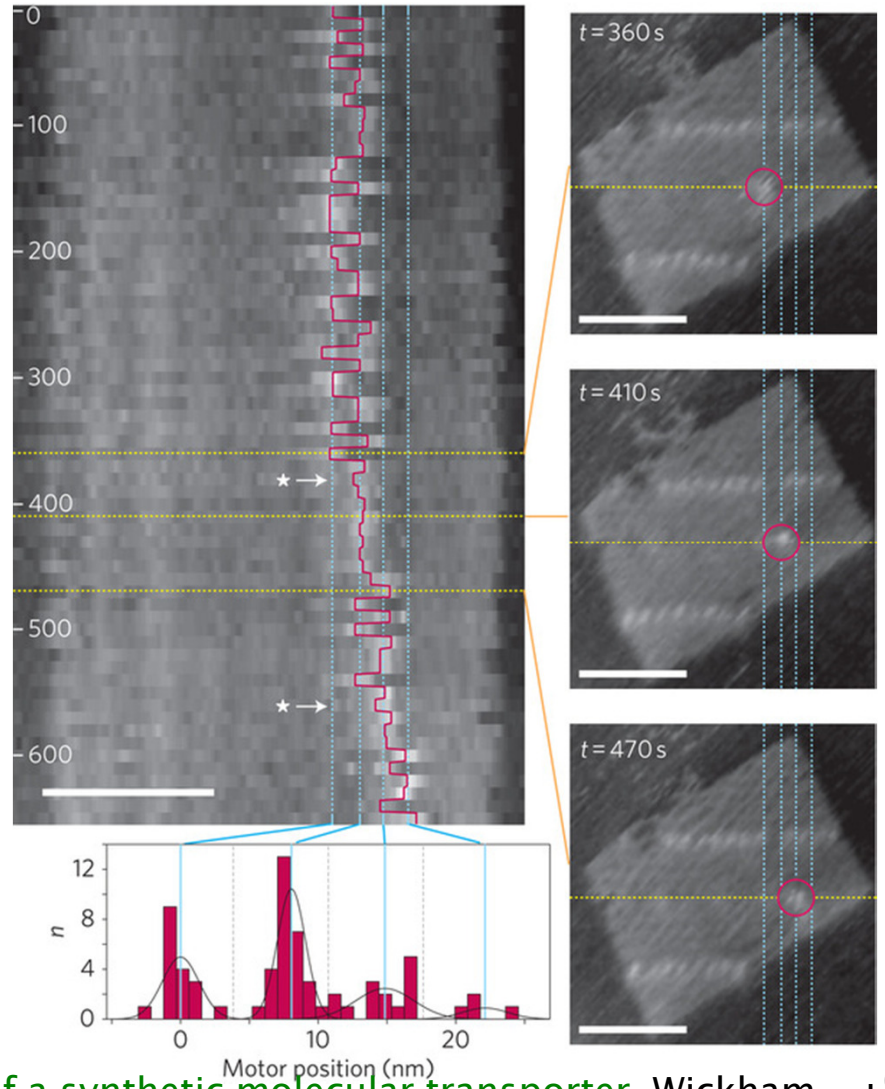
a. Tile design, showing staples 'pinning down' the monomer and highlighting seam staples
b. Circular single strand that folds into tile
c. AFM image of the tile

Guiding the folding pathway of DNA origami. Dunne, Dannenberg, Ouldridge, Kwiatkowska, Turberfield & Bath, Nature (in press)

Video by Christina Furse Davis

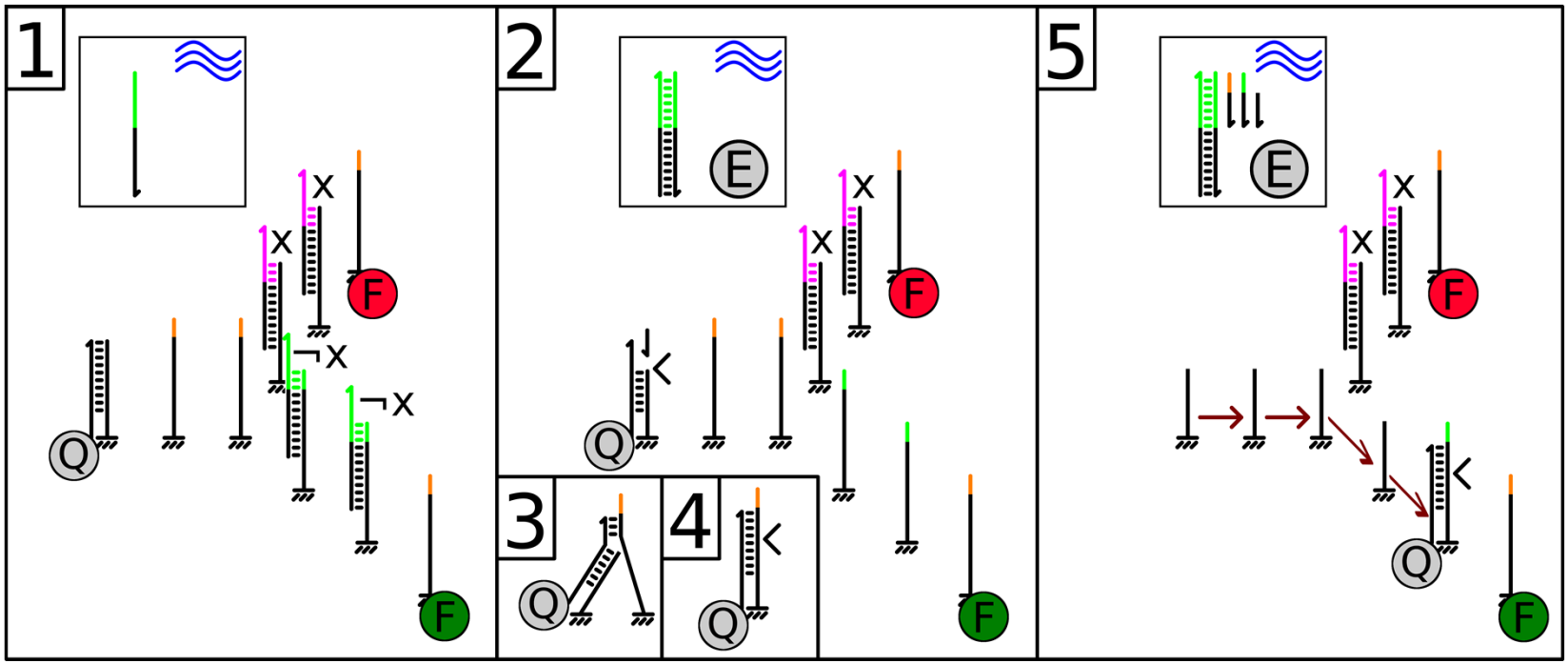
DNA walkers

- How it works…
  - tracks made up of anchor strands laid out on DNA origami tile
  - can make molecule 'walk' by attaching/ detaching from anchor
  - autonomous, constant average speed
  - can control movement
  - can carry cargo
  - all made from DNA



Direct observation of stepwise movement of a synthetic molecular transporter. Wickham *et al*, Nature Nanotechnology 6, 166–169 (2011)

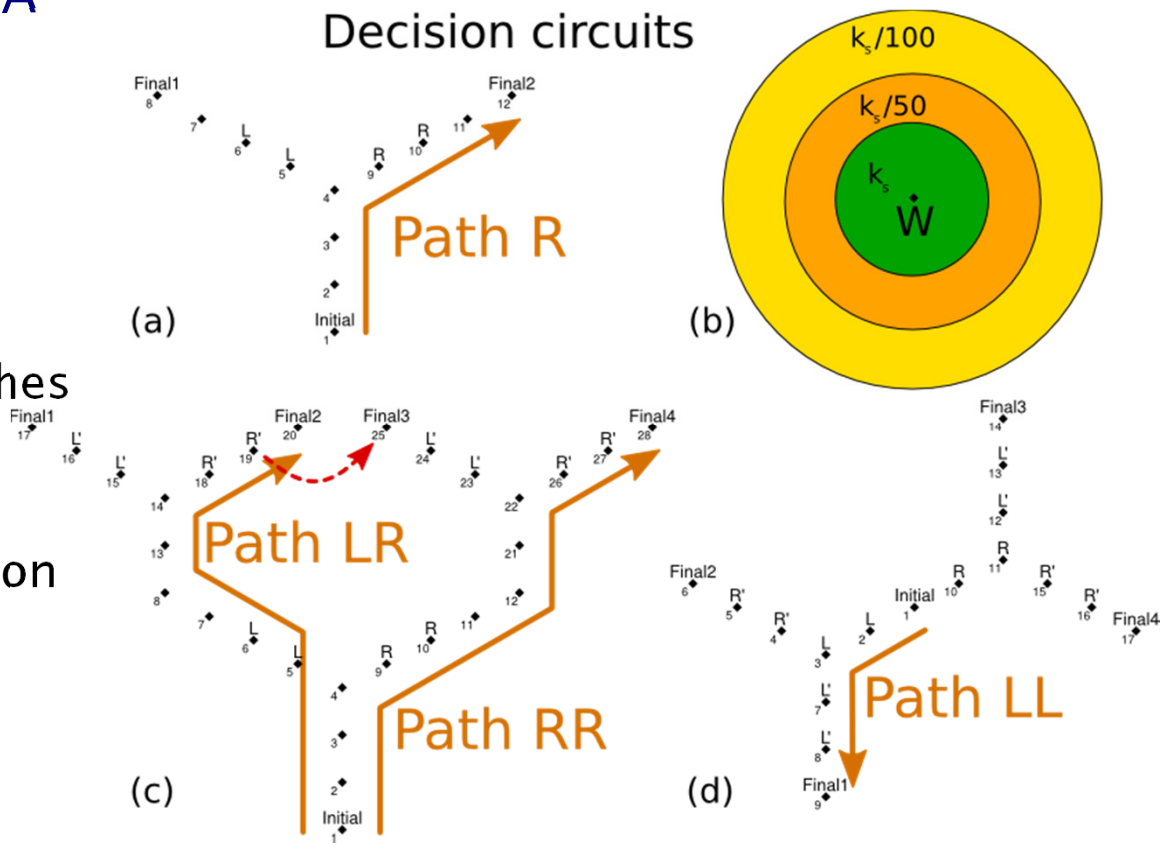
Walker stepping action in detail…



1. Walker carries a quencher (Q)
2. Sections of the track can be selectively unblocked
3. Walker detaches from anchor strand
4. Walker attaches to the next anchor along the track
5. Fluorophores (F) detect walker reaching the end of the track

DNA walker circuits

- Computing with DNA walkers
  - branching tracks laid out on DNA origami tile
  - starts at ‘initial’, signals when reaches ‘final’
  - can control ‘left’/’right’ decision
  - (this technology) single use only, ‘burns’ anchors



- Localised computation, well mixed assumption as in solution does not apply

Why DNA programming?

- DNA: versatile, easily accessible, cheap to synthesise material
- Good for biosensors
  - programmable identification of substance, targeted delivery
- Moore's law, hence need to make devices smaller…
  - DNA computation, directly at the molecular level
  - nanorobotics, via programmable molecular motion
- Many applications for combinations of DNA logic circuits, origami and nanorobotics technologies
  - e.g. point of care diagnostics, smart therapeutics, …

- What good is quantitative verification in this application domain?
  - stochasticity essential!
  - reliability of computation is an issue

This lecture…

- The setting: DNA walker circuits
- Quantitative modelling and verification for molecular programming
  - probabilistic model checking and PRISM
  - automatic debugging DNA computing devices
  - analysing reliability of molecular walkers
  - not just verification: can we automatically synthesise reaction rates to guarantee a specified level of reliability?
- The question: Can we use stochastic Petri nets to model and analyse DNA walker circuits?
  - compare Cosmos with PRISM (thanks to Benoit Barbot)
- Challenges and directions

Modelling frameworks

- Assume wish to model a network of molecules
  - N different molecular species, interact through reactions
  - fixed volume V (spatially uniform), constant pressure and temperature
- Continuous deterministic approach
  - approximate the number of molecules in V at time t by a continuous function, assuming large numbers of molecules
  - obtain ODEs (ordinary differential equations)
  - not for individual runs, but average
- Discrete stochastic approach
  - discrete system evolution, via discrete events for reactions
  - obtain discrete-state stochastic process
- Folklore: can obtain different predictions…

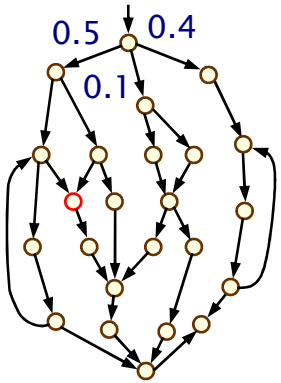
Discrete stochastic approach

- Assume wish to model mixture of molecules
  - N different molecular species, interact through reactions
  - fixed volume V (spatially uniform), constant pressure and temperature
- Work with discrete states as vectors $\underline{x}$ of molecule counts for each species
  - probability $P(\underline{x},t)$ that at time t there will be $\underline{x}_A$ of species A
- Discrete stochastic approach
  - discrete system evolution, via discrete events for reactions
  - essential when molecules in low counts
  - obtain discrete-state stochastic process
  - in fact, if constant state-dependent rates, obtain continuous time Markov chain (CTMC)
- Thus can apply probabilistic model checking techniques…

Quantitative (probabilistic) verification

System

Probabilistic model
e.g. Markov chain



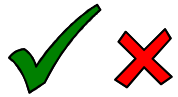
System requirements

$P_{<0.01}$ [$F^{\leq t}$ fail]

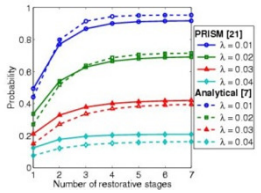
Probabilistic temporal logic specification
e.g. PCTL, CSL, LTL

Probabilistic model checker
PRISM

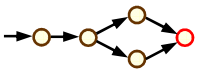
Result



Quantitative results



Counter-example

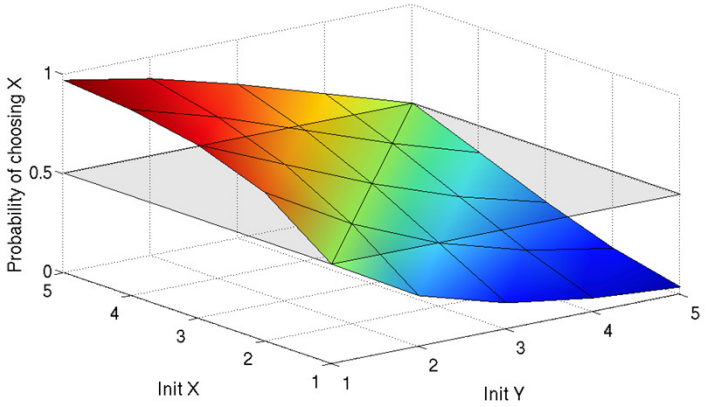


Tool support: PRISM

- PRISM: Probabilistic symbolic model checker
  - developed at Birmingham/Oxford University, since 1999
  - free, open source software (GPL), runs on all major OSs
- Support for:
  - models: DTMCs, CTMCs, MDPs, PTAs, SMGs, …
  - properties: PCTL/PCTL*, CSL, LTL, rPATL, costs/rewards, …
- Features:
  - simple but flexible high-level modelling language
  - user interface: editors, simulator, experiments, graph plotting
  - multiple efficient model checking engines (e.g. symbolic)
- Many import/export options, tool connections
  - MRMC, DSD, Petri nets, Cosmos, Matlab, …
- See: http://www.prismmodelchecker.org/

PRISM – Property specification

- Temporal logic-based property specification language
  - subsumes PCTL, CSL, probabilistic LTL, PCTL*, …
- Simple examples:
  - $P_{\leq 0.01}$ [F "ddl"] – "the probability of deadlock is at most 0.01"
  - $P_{max>0.999}$ [$F^{<10.5}$ "finish"] – "the maximum probability of walker eventually finishing in 10.5 time units is>0.999"
- Usually focus on quantitative (numerical) properties:
  - $P_{=?}$ [F "ddl"]
    "what is the probability of deadlock occurring?"
  - then analyse trends in quantitative properties as system parameters vary



Quantitative probabilistic verification

- What's involved
  - specifying, extracting and building of quantitative models
  - graph-based analysis: reachability + qualitative verification
  - numerical solution, e.g. linear equations/linear programming
  - simulation-based statistical model checking
  - typically computationally more expensive than the non-quantitative case
- The state of the art
  - efficient techniques for a range of probabilistic real-time models
  - feasible for models of up to 10^7 states (10^{10} with symbolic)
  - abstraction refinement (CEGAR) methods
  - multi-objective verification
  - assume-guarantee compositional verification
  - tool support exists and is widely used, e.g. PRISM, MRMC

PRISM – Underlying techniques

- Symbolic implementation
  - data structures based on binary decision diagrams
  - fast construction + compact storage of huge models possible
  - exploit structure, regularity in high-level model
  - usually: up to 10^7–10^8 states; sometimes: up to 10^{10} states
- Numerical solution
  - uniformisation (Jensen's method), for transient probability and rewards
  - fast adaptive uniformisation (FAU), truncates the state space, faster but probability loss
- Simulation-based methods
  - Monte Carlo simulation
  - simulation-based approximate model checking (statistical model checking)

Approximate (statistical) model checking

- Approximate (statistical) probabilistic model checking
  - discrete event (Monte Carlo) simulation + sampling
- Two distinct approaches (both implemented in PRISM)
- Estimation [Hérault et al.]
  - approximate result for quantitative query such as $P_{=?}$ [ϕ]
  - plus a probabilistic guarantee regarding result precision
  - Prob($|p_{actual} - p_{estimated}| \leq \varepsilon$) $\geq 1-\delta$
  - can also generate corresponding confidence intervals
- Hypothesis testing/acceptance sampling [Younes/Simmons]
  - applied to boolean-valued queries such as $P_{\sim p}$ [ϕ]
  - basic idea: stop sampling as soon as the result can be shown to be either true or false with high probability
  - sensitive to distance between bound p and actual answer
  - also extended to Bayesian approaches [Jha et al.]

Approximate (statistical) model checking

- Advantages
  - much more scalable that conventional (numerical computation based) probabilistic model checking
  - (almost no scalability issues – no need to build model)
  - wider range of model types (anything that can be effectively simulated) and property types
- Disadvantages
  - loss of precision: only approximate answers
  - lose ability to definitively establish causal relationships and identify best/worst-case scenarios
  - speed: possibly very high number of samples required to generate suitable accurate approximations
  - may be hard to estimate likelihood of rare events

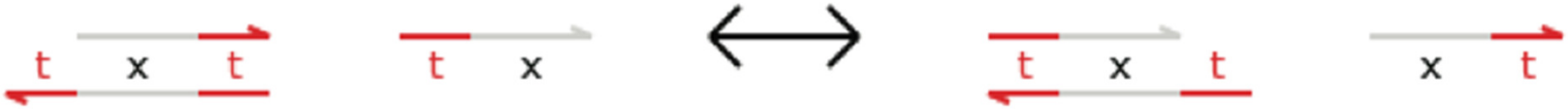
Historical perspective

- First use of PRISM for modelling molecular networks in 2005
  - [Calder, Vyshemirsky, Gilbert and Orton, …]
  - RKIP inhibited ERK pathway
- 2006 onwards: PRISM enhanced with SBML import
  - predictive modelling of the FGF pathway [Heath, Kwiatkowska, Norman, Parker and Tymchyshyn]
  - predictions experimentally validated [Sandilands et al, 2007]
- Since 2012 PRISM has been applied to DNA computation
  - PRISM connected to Microsoft's Visual DSD (DNA computing design tool) [Lakin, Parker, Cardelli, Kwiatkowska and Phillips]
  - expressiveness and reliability of DNA walker circuits studied [Dannenberg, Kwiatkowska, Thachuk, Turberfield]
- Scalability of PRISM analysis limited

Cardelli's DNA transducer gate

- DNA computing with a restricted class of DNA strand displacement structures (process algebra by Cardelli)
  - double strands with nicks (interruptions) in the top strand

  - and two-domain single strands consisting of one toehold domain and one recognition domain



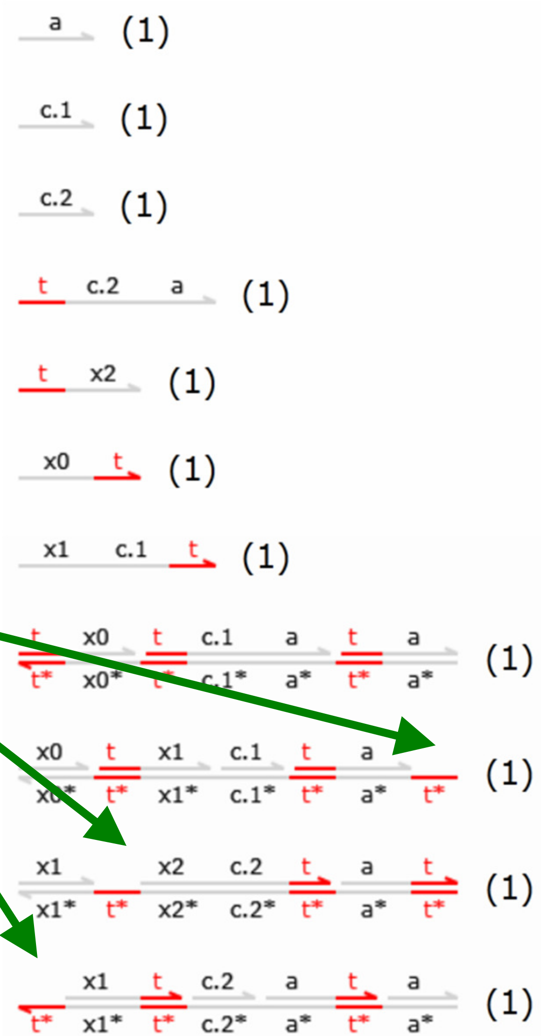
  - "toehold exchange": branch migration of strand <t^ x> leading to displacement of strand <x t^>
- Used to construct transducers, fork/join gates
  - which can emulate Petri net transitions
  - can be formed into cascades [Qian, Winfree, *Science* 2011]

Two-Domain DNA Strand Displacement. Cardelli, L. *Proc. Development of Computational Models (DCM'10)*, 2010

DNA transducer flaw

- Cardelli's DNA transducer gate
  - inputs/outputs single strands
  - can be connected into cascades
- PRISM identifies a bug: 5-step trace to a "bad" deadlock state
  - previously found manually [Cardelli'10]
  - detection now fully automated
- Bug is easily fixed
  - (and verified)

reactive gates



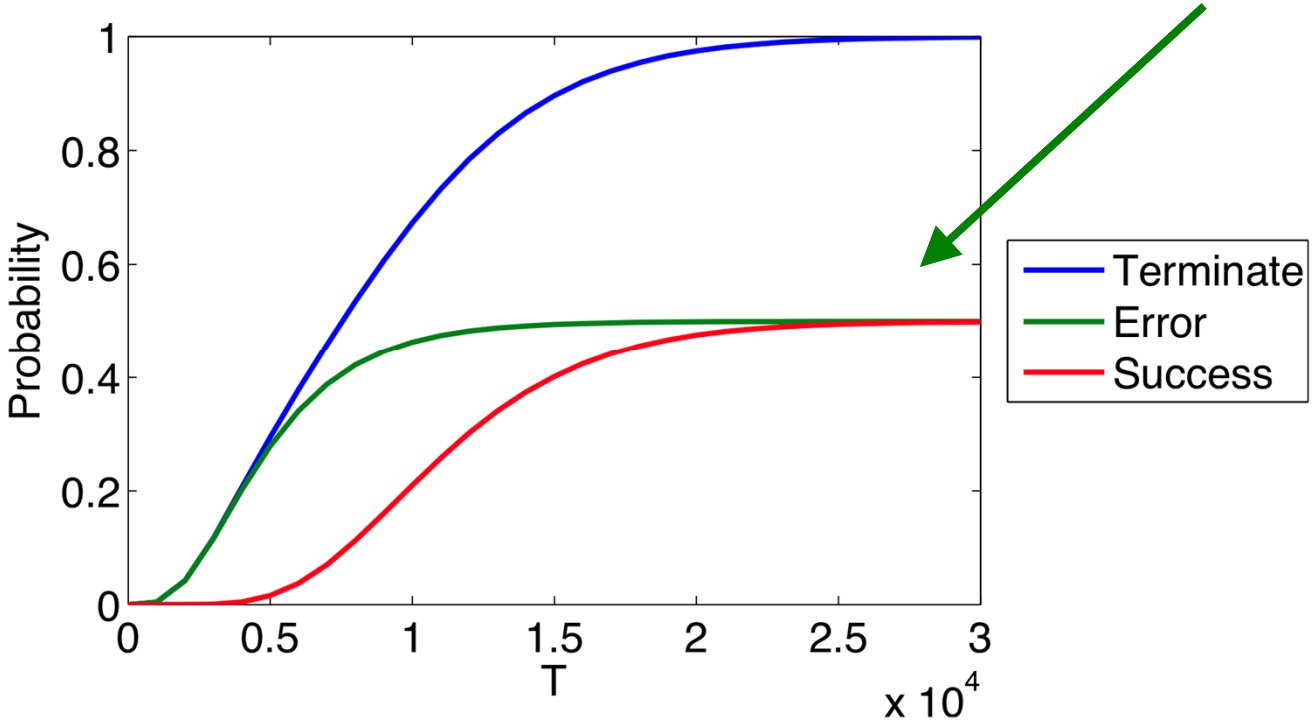
Counterexample:
(1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
(0,1,1,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
(0,0,1,0,1,1,1,1,1,0,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
(0,0,1,0,1,1,1,1,0,0,1,1,1,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
(0,0,1,0,1,1,0,1,0,0,1,1,1,0,0,0,1,0,0,0,0,1,1,1,0,0,0,0,0,0,0,0)
(0,0,1,0,1,1,0,1,0,0,1,0,1,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0)

Design and Analysis of DNA Strand Displacement Devices using Probabilistic Model Checking, Lakin *et al*, Journal of the Royal Society Interface, 9(72), 1470-1485, 2012

Quantitative properties

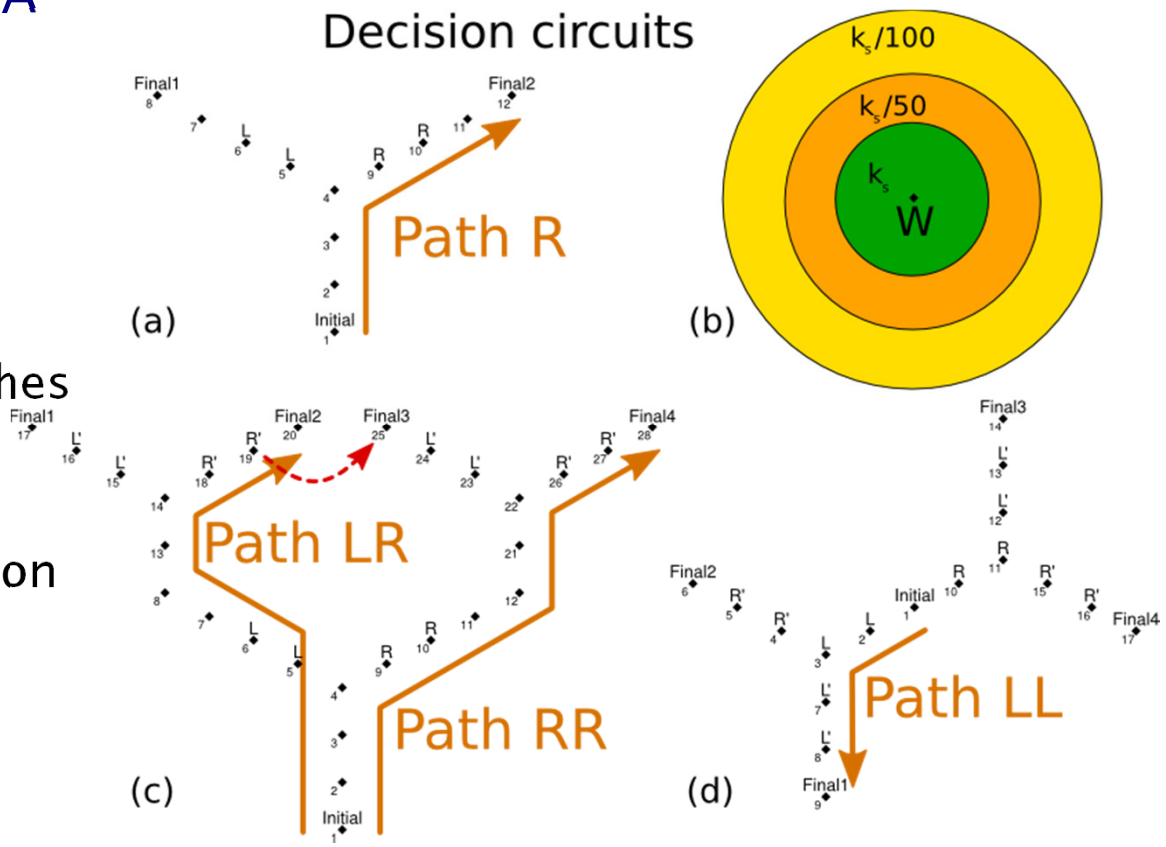
- We can also use PRISM to study the kinetics of the pair of (faulty) transducers:
  - $P_{=?}$ [$F^{[T,T]}$ "deadlock"]
  - $P_{=?}$ [$F^{[T,T]}$ "deadlock" & !"all_done"]
  - $P_{=?}$ [$F^{[T,T]}$ "deadlock" & "all_done"]

success/error equally likely



Recall DNA walker circuits

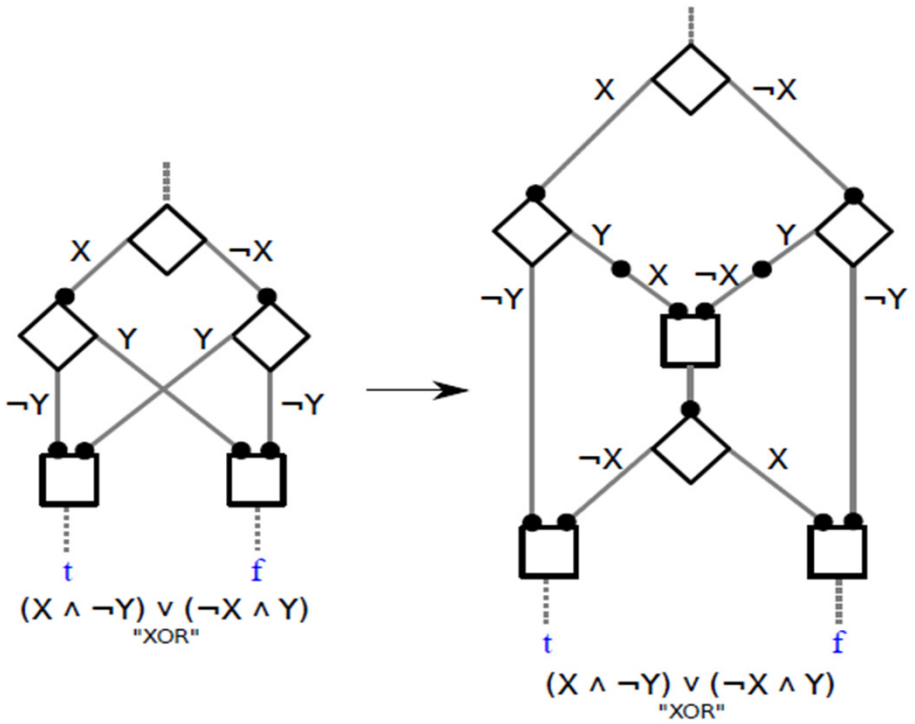
- Computing with DNA walkers
  - branching tracks laid out on DNA origami tile
  - starts at ‘initial’, signals when reaches ‘final’
  - can control ‘left’/’right’ decision
  - (this technology) single use only, ‘burns’ anchors



- But what can they compute?

DNA walkers: expressiveness

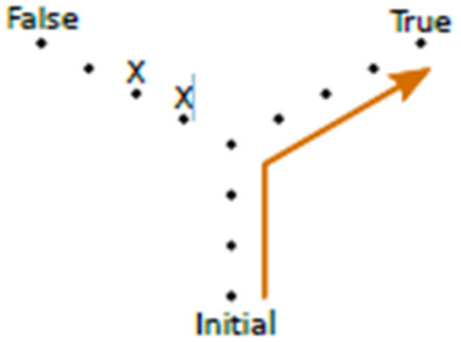
- Several molecular walker technologies exist
  - computation localised
  - faster computation times than in solution
- The 'burnt bridges' DNA walker technology
  - can compute any Boolean function
  - must be planar, needs rerouting
  - tracks undirected
  - reduction to 3-CNF, via a series of disjunction gates
  - limited parallel evaluation

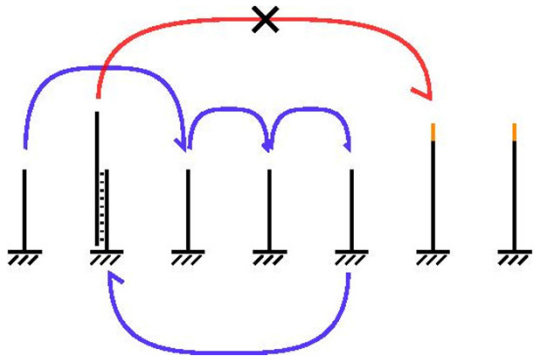


DNA walker circuits: Computational potential, design, and verification. Dannenberg *et al*, Natural Computing, To appear, 2014

DNA walkers: applications

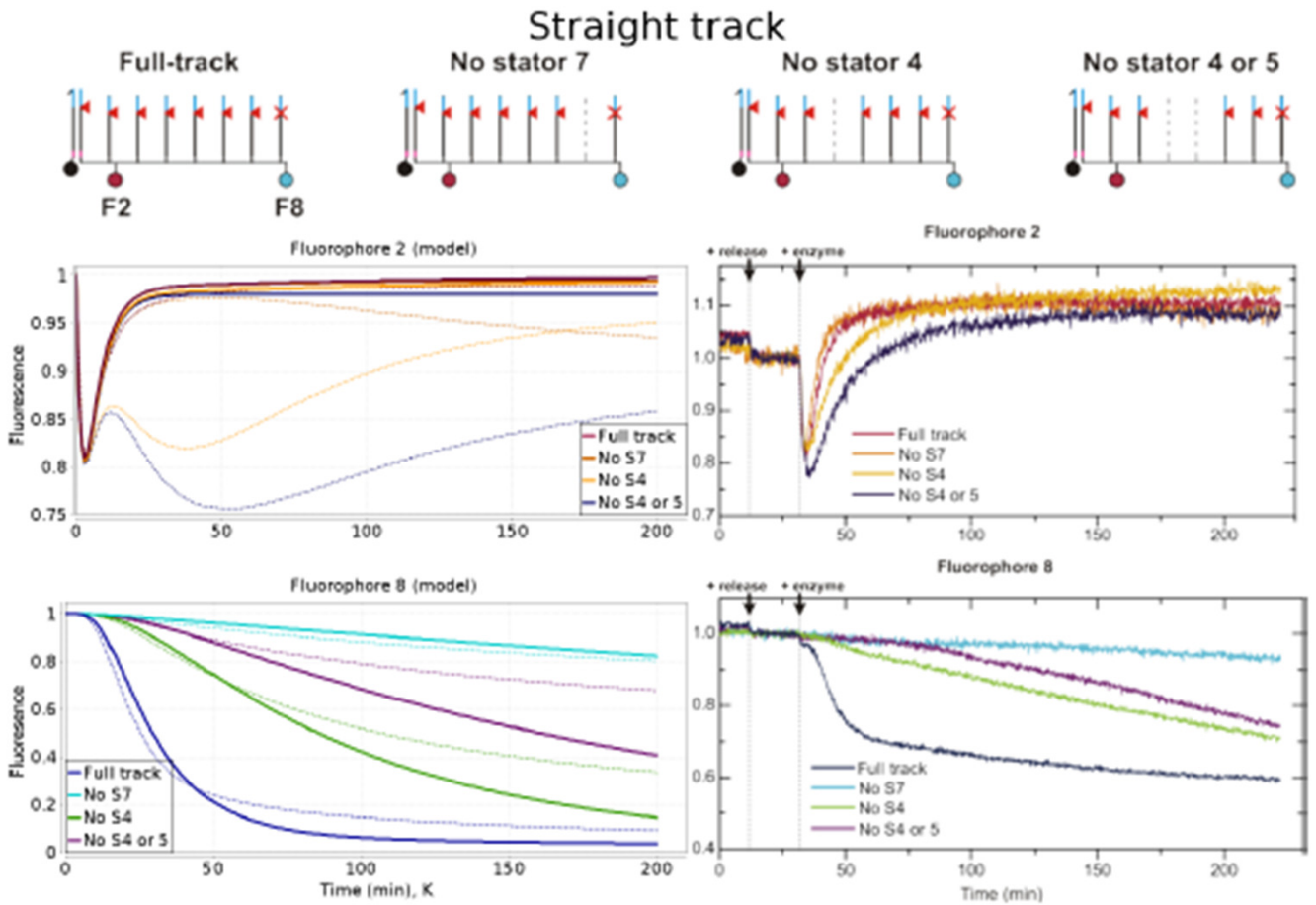
- Walkers can realise biosensors: safety/reliability paramount
- Molecular walker computation inherently unreliable…
  - 87% follow the correct path
  - can jump over one or two anchorages, can deadlock





- Analyse reliability of molecular walker circuits using PRISM
  - devise a CTMC model, fit to experimental data
  - analyse reliability, deadlock and performance
  - use model checking results to improve the layout

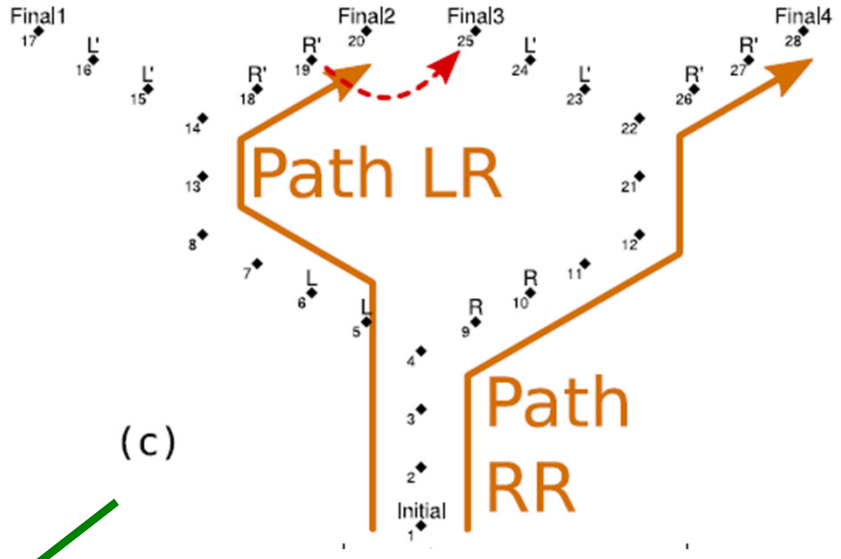
DNA walkers: model fitting



Fitting single-junction circuit to data (dotted lines alternative model)

DNA walkers: results

- Model predictions reasonably well aligned with experiments
- Results confirm effect of leak reactions
- Improve layout guided by model checking
- Can synthesise rates to guarantee reliability level

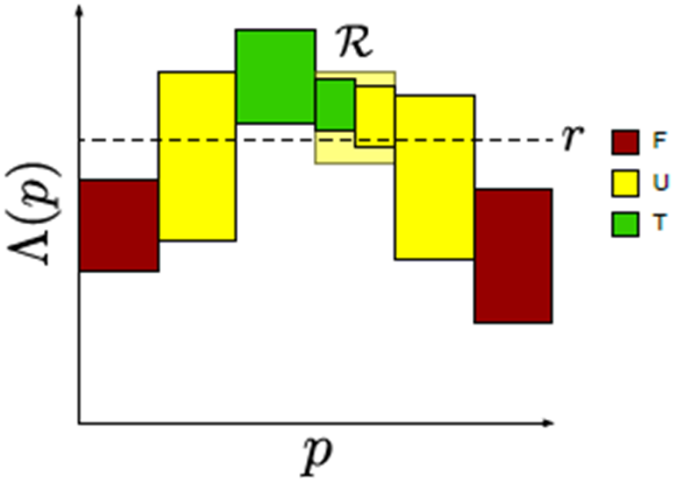


	Single-junction						Double-junction									
	Experiment			Model			Experiment				Model				Optimised (model)			
%	R	R^2	L/R	R	R^2	L/R	RR	RL	LR	LL	RR	RL	LR	LL	RR	RL	LR	LL
Finishes	65	56	56	97	96	92	33	40	22	33	90	89	89	90	94	92	94	92
Correct	76	87	50	78	85	50	70	65	55	76	77	74	74	77	78	78	78	78
Deadlock				.084	.16	.063					1.0	1.7	1.7	1.1	0.0	0.0	0.0	0.0
Steps				7.1	7.0	6.6					11.7	11.8	11.8	11.7	5.1	5.1	5.1	5.1

http://www.prismmodelchecker.org/casestudies/dna_walkers.php

From verification to synthesis…

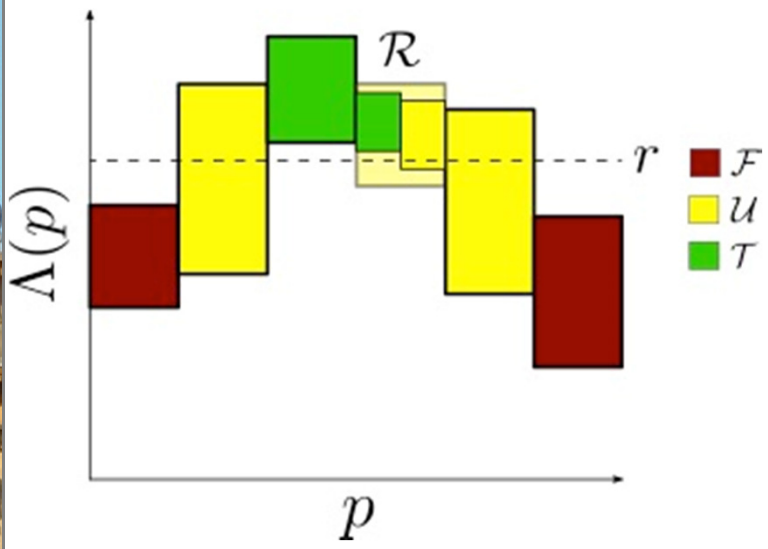
- Automated verification aims to establish if a property holds for a given model
- Can we find a model so that a property is satisfied?
  - difficult…
- The parameter synthesis problem is
  - given a parametric model, property and probability threshold
  - find a partition of the parameter space into True, False and Uncertain regions s.t. the relative volume of Uncertain is less or equal than a given ϵ
- Successive region refinement, based on over & under approx., implemented in PRISM



Precise Parameter Synthesis for Stochastic Biochemical Systems. Ceska *et al*, In Proc. CMSB, LNCS, 2014

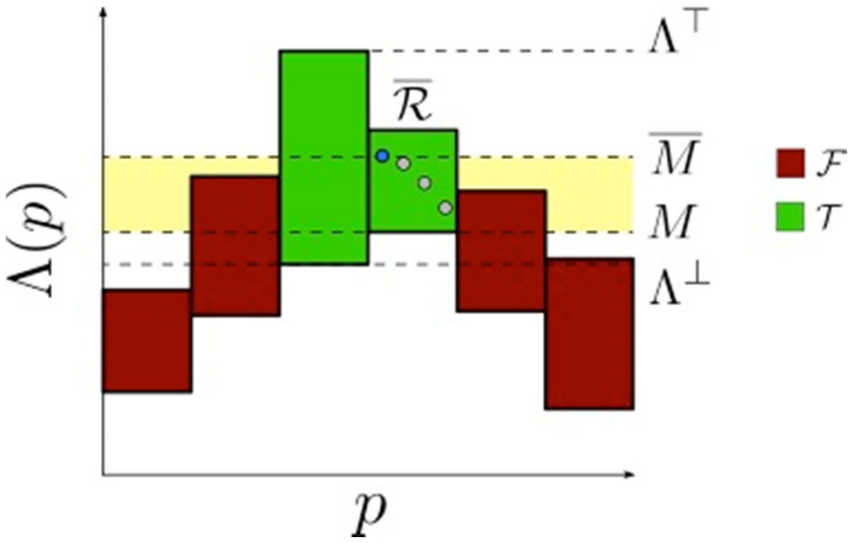
Example: synthesis

Threshold ($\geq r$)



- True if lower bound above r
- False if upper bound below r
- Undecided otherwise (to refine)

Max

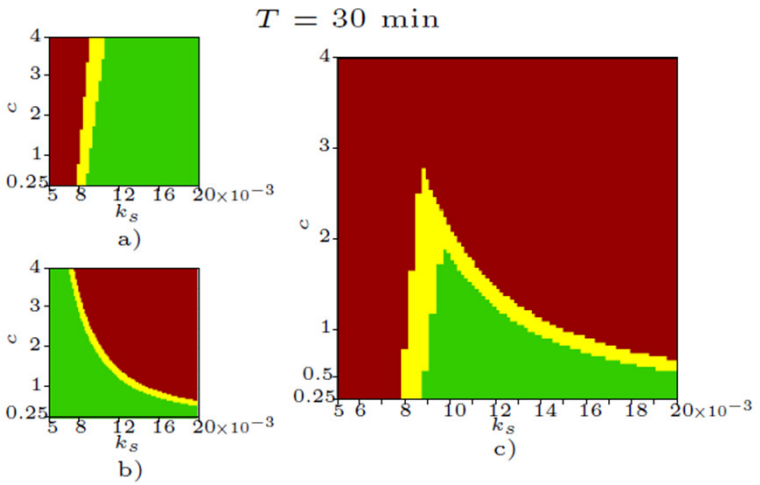


- False if upper bound below under-approximation of max prob M
- True otherwise (to refine)

DNA walkers: parameter synthesis

- Application to biosensor design: can we synthesise the values of rates to guarantee a specified reliability level?
- For the walker model:
  - walker stepping rate k = funct (k_s,c) where k_s lies in interval [0.005,0.020], c in [0.25, 4]
  - find regions of values of k_s and c where property is satisfied

a) $\Phi_1 = P_{\geq 0.4}[F^{[30,30]}$ finish-correct$]$
b) $\Phi_2 = P_{\leq 0.08}[F^{[30,30]}$ finish-incorrect$]$
c) $\Phi_1 \wedge \Phi_2$



- Fast: for T=200, 88s with sampling, 329 subspaces

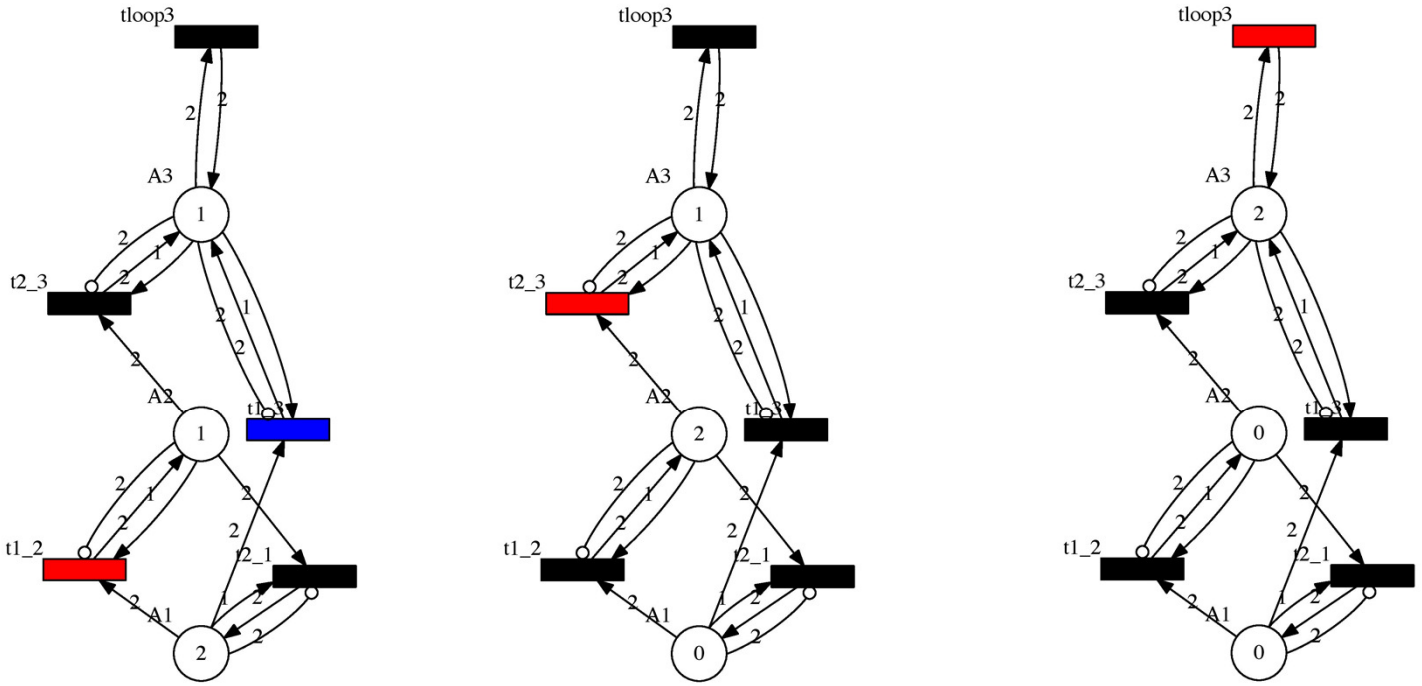
What has been achieved?

- Some successes
  - automatically found a flaw in DNA program
  - design automation for DNA walker circuits, can guarantee reliability levels, fast
- Improved computational performance
  - fast adaptive uniformisation (FAU): significant improvement in computational performance and memory at a cost of precision (but see also adaptive aggregation in [CAV 2015])
  - parameter synthesis: region refinement in conjunction with sampling
- Limited scalability
  - DNA transducer: 6–7 molecules
  - DNA walker circuits: smaller models can be handled with FAU, lager ones only with statistical model checking
  - DNA origami folding: only simulation is feasible

Why (stochastic) Petri nets?

- Excellent match to the problem domain
  - same expressive power as (stochastic) chemical reaction networks
  - more expressive than PRISM's (finite) reactive modules
  - used for modelling of molecular networks since 1990s, e.g. [Goss and Peccoud, PNAS 1998]
- Ease of modelling
  - graphical, facilitates circuit layout
  - walker function modelled as a token game
  - several tools available: Cosmos, MARCIE, ULTRASAN…
- Opportunity to try out Cosmos powerful functionality…
  - GSPN support (immediate transitions)
  - expressive property specification formalism
  - statistical model checking via efficient (parallel) simulation

DNA walker as a Petri net



- Model range of circuit designs, including blockage failure (30%)
- Analyse probability of deadlock, reliability and performance
- Compare against PRISM (uniformisation, FAU and CI statistical model checking)

Results

- Cosmos and PRISM statistical model checking engines
  - indistinguishable results (2m simulations, 0.99 CI)
  - but Cosmos faster, exploits structure of the Petri net in simulation and parallelisation
- PRISM FAU
  - fastest on small models
  - greater memory requirement than statistical methods
- PRISM standard uniformisation
  - suffers from state-space explosion
  - slowest
- Cosmos statistical model checking
  - less precise than FAU on small models due to probab. lost
  - but can be more precise than FAU (CI 0.0018 vs 0.02 lost)
- See tables in the full paper [Barbot, Kwiatkowska 2015]

Conclusions

- Demonstrated that quantitative verification can play a central role in design automation of molecular devices
- Many positive results:
  - bugs found in small scale molecular systems
  - successful experimental validation
  - automatically determined rates that guarantee reliability level
  - demonstrated practical feasibility with good accuracy of statistical model checking
- Key challenge (as always): state space explosion
  - can we exploit compositionality in analysis?
  - can we synthesise walker circuit layout?
  - parameter/model synthesis for more complex models…

Acknowledgements

- My group and collaborators n this work
- Project funding
  - ERC, EPSRC, Microsoft Research
  - Oxford Martin School, Institute for the Future of Computing

- See also
  - VERIWARE www.veriware.org
  - PRISM www.prismmodelchecker.org